

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-235004  
(43)Date of publication of application : 13.09.1996

(51)Int.Cl. G06F 9/46  
G05B 19/02

(21)Application number : 07-205600 (71)Applicant : MITSUBISHI ELECTRIC CORP  
(22)Date of filing : 11.08.1995 (72)Inventor : NAMIKADO SHIGEKI

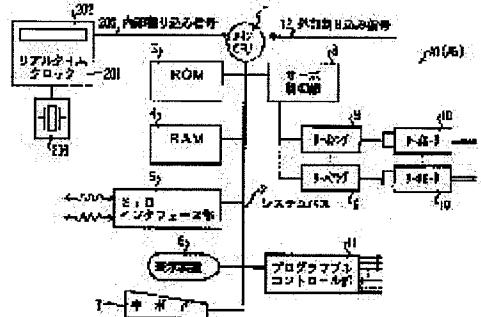
(30)Priority  
Priority number : 06323233 Priority date : 26.12.1994 Priority country : JP

## (54) CONTROL METHOD FOR CONTROL SOFTWARE EXECUTION SYSTEM

(57)Abstract:

**PURPOSE:** To effectively control a control software execution system by securing the flexibility for an execution designation system of every task and then setting a proper processing time to every task.

**CONSTITUTION:** In this control system, a scheduler is provided in a real-time operating system included in a memory to perform the switching processing of different tasks. Then the execution of plural tasks which execute the control software is controlled based on the task switching control of the scheduler. In such a constitution of the system, the time passed from the present time point is calculated to assure that every task runs at each prescribed unit time that is previously designated. This calculated passed time is written in a counter 202, and the task is started when the passed time written in the counter 202 elapses. In other words, a function is secured to make every task run in a cycle that is previously designated. Thus it is possible to control the periodical start of all tasks in a small time unit and furthermore to easily designate the task starting cycle.



(19)日本国特許庁 (JP)

(12) **公開特許公報 (A)**

(11)特許出願公開番号

**特開平8-235004**

(43)公開日 平成8年(1996)9月13日

(51)Int.Cl. <sup>6</sup>	識別記号	府内整理番号	F I	技術表示箇所
G 0 6 F 9/46	3 4 0		G 0 6 F 9/46	3 4 0 E
G 0 5 B 19/02			G 0 5 B 19/02	T

審査請求 未請求 請求項の数29 O.L (全114頁)

(21)出願番号	特願平7-205600	(71)出願人	000006013 三菱電機株式会社 東京都千代田区丸の内二丁目2番3号
(22)出願日	平成7年(1995)8月11日	(72)発明者	南角 茂樹 東京都千代田区丸の内二丁目2番3号 三菱電機株式会社内
(31)優先権主張番号	特願平6-323233	(74)代理人	弁理士 酒井 宏明
(32)優先日	平6(1994)12月26日		
(33)優先権主張国	日本 (JP)		

(54)【発明の名称】 制御ソフトウェア実行システムの制御方法

(57)【要約】

【目的】 各タスクの実行の指定方式に柔軟性を持たせることによって各タスクに適切な処理時間を与え、そのことにより、効率のよい制御ソフトウェア実行システムの制御方法を得る。

【構成】 リアルタイムオペレーティングシステムがタスク単位で予め指定した所定の単位時間毎に走ることを保証するために現在からの経過時間を計算し、該計算された経過時間をカウンタに書き込み、該カウンタに書き込まれた経過時間の経過後に当該タスクを走行させる。

## 【特許請求の範囲】

【請求項 1】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 2】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 3】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間で連続して走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 4】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に、予め指定した割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 5】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記

システムが、タスク毎に予め指定した所定の単位時間毎に、予め指定した所定の単位時間で連続して走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 6】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、タスクスイッチの発生時にカウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値とを記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 7】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、タスクスイッチの発生時にカウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値とを記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 8】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、タスクや割り込み処理の走行時間を計測することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項 9】 メモリ内部のリアルタイムオペレーティ

ングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、運転条件により前記走行割合を変更することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項10】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項11】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定することを

特徴とする制御ソフトウェア実行システムの制御方法。

【請求項12】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定し、前記タスクが工具に影響を受けるタスクか否かを判定し、工具に影響を受けるタスクであると判定した場合は別の工具を選択し、前記選択された工具によって指定された加工プログラムを再実行することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項13】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定し、前記タスクが工具の送り速度の影響を受けるタスクか否かを判定し、工具の送り速度の影響を受けるタスク

5 であると判定した場合は別の送り速度を選択し、前記選択された工具の送り速度によって指定された加工プログラムを再実行することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項14】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定し、前記タスクが工具の送り速度の影響を受けるタスクか否かを判定し、工具の送り速度の影響を受けるタスクであると判定した場合は別の送り速度を選択し、前記選択された工具の送り速度によって指定された加工プログラムを再実行することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項15】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間を前記カウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを中断させてタスクの走行時間を制限することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項16】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウ

ンタに書き込まれた経過時間の経過後に前記タスクを停止させ、前記制御ソフトウェアを複数ブロック解析することにより、割り当てられた時間の範囲内で、複数のブロックの処理を行なうことの特徴とする制御ソフトウェア実行システムの制御方法。

【請求項17】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記計算された基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

10 【請求項18】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを所定の割合走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

20 【請求項19】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを所定の割合走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

30 【請求項20】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の時間連続して走ることを保証するために、現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記所定時間前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

40 【請求項21】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に、予め指定した割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間

7  
が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項21】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク毎に予め指定した所定の時間毎に、予め指定した所定の単位時間は、連続して走ることを保証するために、現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを所定時間連続して走行させることを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項22】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムの基準クロックを記録し、前記タスクの実行権がなくなったときの前記記録手段のデータから前記タスクが走行した時間を計算し、前記タスクの走行時間の合計を計算し、前記タスクの合計走行時間を記録することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項23】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムの基準クロックを記録し、前記タスクの実行権がなくなったときの前記システムの基準クロックのデータから前記タスクが走行した時間を計算し、前記タスクの走行時間の合計を計算し、割り込み発生時の割り込み処理も優先順位の高いタスクによって計算し、前記タスクの合計走行時間および割り込み処理の走行時間を記録することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項24】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロック

の何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記タスク毎に基準クロック数を記録し、前記経過時間の経過後に前記タスクを走行させ、前記タスクの実行権が移動するたびに前記基準クロック数を加算し、前記経過時間の経過後に前記タスクを所定の割合走行させると共にタスクや割り込み処理の走行時間を計測することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項25】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが、前記基準クロック数分経過したことを判断し、運転条件により前記走行割合を変更することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項26】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記タスクスイッチの発生時に、タスク毎に経過クロック数を読み込み、前記タスク毎に備えた前記クロック数の合計を記録し、前記クロック数経過後に前記タスクを走行させるとともに、運転を実行しながら自動的に各タスクの走行時間の割合を変更することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項27】 メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過クロック数を計算し、前記クロック数を記録し、前記クロック数の経過後に前記タスクを走行させ、タスクスイッチの発生時に、タスク毎に経過クロック数を読み込み、タスク毎に備えた前記経過クロック数の合計を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を

指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定し、運転を行いながらタスクの異常事態を判定することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項28】メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過クロック数を計算し、前記クロック数を記録し、前記クロック数の経過後に前記タスクを中断し、前記タスクの走行時間を制限することを特徴とする制御ソフトウェア実行システムの制御方法。

【請求項29】メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過クロック数を計算し、前記クロック数の経過後に前記タスクを停止し、前記制御ソフトウェアを複数ブロック解析することにより、割り当てられた時間の範囲内で、複数のブロックの処理を行なうことを特徴とする制御ソフトウェア実行システムの制御方法。

#### 【発明の詳細な説明】

##### 【0001】

【発明の属する技術分野】この発明は、制御ソフトウェア実行システムの制御方法に関し、さらに詳細には、数値制御装置（以下、NC装置）の制御用ソフトウェア（コントロールプログラム）を効率的に実行するNC装置の制御ソフトウェア実行システムの制御方法に関するものである。

##### 【0002】

【従来の技術】図73は、従来におけるNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図であり、図において、501はNC装置全体を示す。以下、このNC装置501のハードウェア構成例を用いて従来におけるNC装置の一般的概略構成を説明する。

【0003】図73において、1は各部の動作を制御したり、数値制御に必要な演算を実行するメインCPU、2は装置各部を結ぶシステムバス、3はNC装置501の主要機能を実現するための制御ソフトウェアなどを格納するROM（不揮発性記憶装置）、4は一時記憶やワークエリアなどに用いられるRAM（揮発性記憶装置）である。

【0004】また、5は外部との間においてシリアル通信によりデータのやり取りを行なうSIOインターフェース部、6はNC装置501の運転状態を表示したり、NC装置501に与えた指令を確認するための表示装置、7はNC装置501に対して指令を与えるためのキーボード（入力装置）、8はサーボモータを制御するための指令を演算するサーボ制御部である。このサーボ制御部8にあってはメインCPU1とは別の専用CPUを備えてもよい。

10 【0005】また、9はサーボ制御部8から受け取った指令を電力増幅してサーボモータや工作機械主軸（図示せず）を駆動するサーボアンプ、10は工作機械（図示せず）の加工部を制御するためのサーボモータ、11は工作機械との間でサーボ制御指令以外のデータをやり取りするためのプログラマブルコントロール部（以下、PC部という）、12はメインCPU1に対して入力されるシステムクロック（図示せず）と外部割り込み信号を表している。システムクロックはNC装置501全体を制御するために同期をとるためのクロック信号である。また、外部割り込み信号12は電源異常や非常停止などイベント（緊急の出来事）の発生をメインCPU1に対して通知するための信号である。

20 【0006】次に、動作について説明する。図74は、図73に示したNC装置501の制御ソフトウェア実行システムの加工プログラム解析実行手順を示す説明図であり、まず、メインCPU1は、ROM3に書き込まれている制御用のソフトウェアをシステムバス2を介して順次1命令ずつ読み込んで、その内容を実行する。

【0007】図74において、加工プログラム入力処理30 21では、SIOインターフェース部5を介して外部から加工プログラム20を読み込み、RAM4に格納する。その後、加工プログラム20をブロック（入力する加工プログラムにより決まっている所定の単位）毎に内部で処理し易いような内部データに変換する。

【0008】また、補正計算処理22では、ブロック毎の内部データを処理、演算して移動量を算出する。また、工具径や工具長などの補正を行なう。さらに、内部座標値の更新処理なども行なう。設定表示処理23では、NC装置501の各種データを表示装置6に表示する。また、キーボード7を用いてオペレータが入力した各種設定データをRAM4に格納する。加えて、補間処理24では、補正計算処理22の処理結果を用いて微小時間毎の各軸の移動量を算出する。

【0009】また、サーボ処理25では、補間処理24の処理結果を、さらに、小さな単位時間毎の各軸の移動量に変換する。さらに、サーボモータ10や工作機械（図示せず）に取り付けている速度検出器および位置検出器（いずれも図示せず）からのフィードバック制御（図示せず）を実行する。プログラマブルコントローラ（以下、PCという）処理26では、工作機械との間に

おいて入出力処理や主軸の制御など、工作機械周辺の制御を実行する。

【0010】さて、NC装置501は、図73を用いて先に説明したようにメインCPU1に外部割り込み信号12が入力し、割り込み処理を実行させることで、非常停止などの通常処理の流れ以外の緊急事態に対処することができる。メインCPU1は、外部割り込み信号12(図73参照)が入力されると、予め指定された別処理を実行し、それら別処理が終了した後に通常の命令に復帰する。

【0011】図75は、割り込み処理を示す概念図である。30～36は通常の命令、37～40は割り込み命令、また、41は通常の命令への復帰命令である。例えば、メインCPU1が通常の命令33を実行中に外部割り込み信号12(図73参照)が入力されると、メインCPU1は、通常の命令33の処理終了後、割り込みを検出し、予め指定されていた割り込み命令37の実行を開始する。その後、割り込み命令37～40の実行を終了した後、復帰命令41を実行して、通常の命令34に復帰し、続いて通常の命令35、36を実行する。

【0012】なお、割り込み処理の時間が長くなると、割り込み処理の実行中に割り込みが再び入ってくるなど多重割り込みの発生する割合が多くなり、また、割り込み処理中は割り込みを禁止していると割り込みに対して迅速に反応できなくなるなどの問題点があるため、割り込み処理は可能な限り短くすることが望ましい。

【0013】ところで、NC装置501の制御ソフトウェアは、以下に示すような特徴を有する。すなわち、第1に、NC装置501の制御ソフトウェアにより実現すべき機能の多様性のため、ソフトウェアも大量となり、多人数で開発される。

【0014】第2に、NC装置501の制御ソフトウェアの機能ごとに、処理に必要な応答時間(ターンアラウンドタイム、デッドライン)が異なる。例えば、サーボ処理25(図74参照)は、処理結果の算出が遅れると切削が止まり、被加工物が不良品になってしまうので、実時間処理でなくてはならない。一方、表示装置6への表示処理などは、多少遅れても不都合は生じないので、実時間処理でなくともよい。

【0015】第3に、NC装置501の制御ソフトウェアの処理は、バッチ処理型あるいはコンパイラ型(1つの加工プログラムを最初から最後まで全て解析し終わってからサーボの移動データを全て一度に出力する)ではなく、インタプリタ型(加工プログラムを、いくつかに分解して、例えば、ブロックに分解し、そのブロック毎に解析していく、サーボの移動データも少しづつ出力する)である。

【0016】第4に、メインCPU1が実行しなくてはならない、割り込み処理は、多くの種類がある。

【0017】これらの特徴により、NC装置501の制

御ソフトウェアは、一般にリアルタイムオペレーティングシステム(以下、RTOSという)を用い、その制御のもとで実行される機能ごとのタスクを実行単位とすることが多い。

【0018】次に、RTOSにより各タスクを制御する方法について説明する。優先順位を一定間隔毎に調べたり、あるタスクを指定時間だけ実行を遅らせたりするために、通常はメインCPU1(図73参照)に対してある一定周期でタイマ割り込み(図示せず)を入れる。これをシステムクロックという。

【0019】また、RTOSは、システムクロック割り込みがある度に各タスクの状態を調べ、実行中のタスクを止めて、別のタスクを実行させたりする。これをスケジューリングあるいはディスパッチという。

【0020】さらに、RTOSにおいては、各タスクに優先順位(実行の優先度)付けを実行する。この優先順位の意味は、より低い優先順位のタスクを実行中に、より高い優先順位のタスクを実行する準備ができたとき、その優先順位の低いタスクの実行を中断させて優先順位の高いタスクを実行させることをいう。これをタスクのプリエムプト(横取り)という。

【0021】図76は、各タスクにおける動作の時間的関係を示すタイミングムチャートであり、図において、特に、補正計算処理タスクなどは、計算対象タスクの複雑さによって処理時間が大きく変化する。通常は、最も処理時間がかかる場合を想定してシステムクロックの同期を決定する。その結果、それ以外の処理を実行する場合には、図76に示したT1のようなメインCPU1が有効な処理を何ら実行していないアイドルタイムが発生する。なお、図76において、他の処理がないときは後述のようにRTOS中のループにおいて処理が待機するものである。

【0022】通常、RTOSは、指定した周期でタスクを走行させたり、タスクの走行時間を測定する機能を有するが、それらはすべてシステムクロック $\Delta T$ の整数倍の単位でしか制御測定することができない。

【0023】また、図76に示した3つのタスクの内、サーボ処理タスク、表示設定処理タスクの処理時間(演算時間)は通常一定である。しかし、補正計算処理タスクは加工プログラムの内容によって処理時間が大きく変化する。一方、サーボ処理タスクは、サーボアンプ9に対するデータ通信等を実行しているため、必ずシステムクロック1回につき1回走行する。このように制御しなければ、アンプユニットの方にサーボモータ10を動作させるデータを送信することができず、サーボモータ10が停止してしまうからである。

【0024】そのため、例えば、補正計算処理タスクを加工プログラム1ブロック毎に終了させて、1回に走行する時間が余り長くならないように制御しているものである。しかし、逆に、補正計算処理タスクの処理時間が

短くて、3つのタスクの処理時間の合計がシステムクロックの $1/N$ （Nは整数）のときであってもシステムクロック1回につき各タスクが1回づつしか走行できない。この制限によってNC装置の切削送り速度の上限が決定されている。

【0025】したがって、例えば、システムクロック1回に各タスクが2回づつ走行することができるとして、切削送り速度を2倍にすることができ、加工時間は、図77に示すように約 $1/2$ となる。

【0026】以上のRTOSのもので、制御ソフトウェアを実行するNC装置501は、何種類かのモードを持っている。例えば、実際に自動運転により被加工物を切削している状態や、手動モードで切削している状態、さらには実加工は行なわずグラフィックによりツールパスを表示させて、加工プログラムの正当性のチェックをしている場合などである。

【0027】また、加工プログラムの入力手段としては、NC装置501の標準の加工プログラム形式である、EIAフォーマットのほかにも、例えば、自動プログラミングというものがある。これは、例えば、使用工具と、素材形状、そして最終の加工形状などを入力する、その結果、図78(a), (b)に示すような、その最終形状を加工するように、工具の速度や位置、動作などを自動的に制御するものである。

【0028】上記のような、EIAフォーマットや、自動プログラミングなどの加工プログラムを解析するタスクは、図79に示すような処理を実行する。すなわち、ZZZ1は、加工プログラムを1ブロック読み出して、メモリ上のワークエリアへ格納する処理である。ZZZ2は、ZZZ1で読み出したデータを解析し、補正計算処理タスクへ渡すデータを作成する処理である。

【0029】ZZZ3は、タスク終了のSVCを発行する処理である。このSVCにより、このタスクは他のタスクにメインCPU1の使用権を譲り渡す。その後、例えば、補正計算処理タスクが、次のデータを要求し、この加工プログラムが解析タスクを起動するまで、システムは待機状態となる。

### 【0030】

【発明が解決しようとする課題】しかしながら、上記に示した構成のNC装置の制御ソフトウェア実行システムにあっては、その制御ソフトウェアを実行するとき、以下のようないくつかの問題点が発生する。

【0031】第1に、NC装置としての機能を実現する各タスクは、優先順位によってしかその実行の順番が決められない、そのため全てのタスクを周期的に走らせたり、タスク毎に実行時間を決めたりすることができない。

【0032】第2に、タスクの制御を時間により実行するような場合であっても、その単位はシステムクロックの単位（システムクロックの整数倍）に限定される。

【0033】第3に、タスクの優先順位の決め方が難しく、特に、3個以上のタスクが関係すると、いわゆるプライオリティインバージョン（優先順位の低いタスクが優先順位の高いタスクを止めてしまうような優先順位の逆転現象）が発生する可能性があるため、優先順位だけによる実行方式には限界がある。

【0034】第4に、システムの生成時にタスク間の優先順位を定めてしまうため、処理時間が足りないタスクや、逆に処理時間が余るタスクが存在する。

10 10 【0035】■第5に、周期を指定したい処理は割り込み処理としなくてはならないため、その処理時間を短くしなくてはならないなどの制限がある。

【0036】第6に、NC装置の運転状態や、モードによって各タスクにかかる負荷は一定ではないにもかかわらず、各タスクの走行状態をシステム生成時に決定して、そのまま変更できない。

【0037】第7に、あるタスクが正常ではない（異常状態に陥った）か否かを判定する手段がない。

20 20 【0038】第8に、加工プログラムのブロックの大きさにかかわらず、加工プログラム解析タスクが、1ブロック毎に終了するようになっているので、1ブロックの大きさによっては、処理時間が足りなかつたり、逆に余つたりする場合がある。

【0039】この発明は、上記のような問題点を解決するためになされたもので、NC装置における各タスクの実行の指定方式に柔軟性を持たせることによって各タスクに適切な処理時間を与え、そのことにより、効率のよい制御ソフトウェア実行システムの制御方法を得ることを目的とする。

30 30 【0040】

【課題を解決するための手段】この発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させるものである。すなわち、個々のタスクを予め指定した周期で走らせる機能を有するものである。

40 40 【0041】したがって、全てのタスクを周期的に起動する、例えば、NC装置機能を実現するタスクの時間による制御を小さな単位で行い、さらにタスクを起動する周期を容易に指定することができる。

【0042】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチ

グ処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させるものである。すなわち、個々のタスクを指定した割合走らせる機能を有するものである。

【0043】したがって、全てのタスクの走行時間の割合を指定でき、例えば、NC装置機能を実現するタスクの走行時間による制御を実行することができる。

【0044】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間で連続して走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させるものである。すなわち、個々のタスクの指定した時間連続して走らせる機能を有するものである。

【0045】したがって、全てのタスクの連続走行時間を指定でき、例えば、NC装置機能を実現するタスクに連続した走行時間を保証することができる。

【0046】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に、予め指定した割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させるものである。すなわち、個々のタスクを予め指定した周期で、指定した割合走らせる機能を有するものである。

【0047】したがって、全てのタスクを周期的に起動し、さらに走行時間の割合を指定でき、例えば、NC装置機能を実現するタスクの正確な起動時間と走行時間による制御を実行することができる。

【0048】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェ

アを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク毎に予め指定した所定の単位時間毎に、予め指定した所定の単位時間で連続して走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させるものである。すなわち、個々のタスクを予め指定した周期で、指定した時間連続して走らせる機能を有するものである。

10 【0049】したがって、全てのタスクを周期的に起動し、さらに連続した走行時間を指定でき、例えば、NC装置機能を実現するタスクの正確な起動時間と連続走行時間による制御を実行することができる。

【0050】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、タスクスイッチの発生時にカウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値とを記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録するものである。すなわち、前記各タスクの走行時間を計測、記録する機能を有するものである。

【0051】したがって、全てのタスクの走行時間を記録し、例えば、NC装置機能を実現するタスクの正確な実行時間を知ることができる。

【0052】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、タスクスイッチの発生時にカウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値とを記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理するものである。すなわち、割り込み処理もタスクとして実行してその走行時間も計測、記録する機能を有するものである。

40 【0053】したがって、割り込み処理の走行時間を記録し、例えば、NC装置機能を実現するタスクと割り込み処理の正確な実行時間を知ることができる。

【0054】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオ

ペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、タスクや割り込み処理の走行時間を計測するものである。すなわち、前記各タスクの走行時間を計測、記録する機能と個々のタスクを指定した割合走らせる機能を有するものである。

【0055】したがって、全てのタスクの走行時間を記録し、さらに全てのタスクの走行時間の割合を指定する。例えば、NC装置機能を実現するタスクと、割り込み処理の正確な実行時間を知ることができる。

【0056】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、運転条件により前記走行割合を変更するものである。すなわち、例えば、NC装置の実行している運転状態により各タスクの走行時間の割合を変更する機能を有するものである。

【0057】したがって、装置の運転状態により、各タスクへの走行時間の割り当てを自動的に最適なものにし、各タスクにおける無駄時間なくし、例えば、NC装置の運転モードによってタスクの走行割合を変化させることができる。

【0058】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前

記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理するものである。すなわち、例えば、NC装置の実行している状態により各タスクの走行時間の割合を、自動的に変更する機能を有するものである。

【0059】したがって、例えば、NC装置を運転しながら、各タスクに割り当てる走行時間の割合を変化させ、各タスクにおける無駄時間がなくし、機能を実現する各タスクへの走行時間の割り当てを、自動的に最適なものにすることができます。

【0060】また、次の発明に係るNC装置の制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定するものである。すなわち、予め指定されたタスクの予想全走行時間と実際の走行時間を比較し、タスクの異常状態を判定するものである。

【0061】したがって、走行時間の比較により異常が発生したタスクを特定でき、例えば、NC装置の機能を実現しているタスクに異常が発生したとき、それを検知することができる。

【0062】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法にお

いて、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常状態を判定し、前記タスクが工具に影響を受けるタスクか否かを判定し、工具に影響を受けるタスクであると判定した場合は別の工具を選択し、前記選択された工具によって指定された加工プログラムを再実行するものである。すなわち、タスクの異常状態を判定し、その異常状態となったタスクが工具の影響を受けるタスクか否かを判定し、その判定結果に基づいて工具を変えて再実行させるものである。

【0063】したがって、例えば、NC装置の稼働中に異常が発生しても、自動的に工具を変えて再実行し、NC装置の自動加工によって、エラーが発生したと判定したときは、工具を変えて再度自動加工を実行することができる。

【0064】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行

時間と実際の走行時間との比較によってタスクの異常状態を判定し、前記タスクが工具の送り速度の影響を受けるタスクか否かを判定し、工具の送り速度の影響を受けるタスクであると判定した場合は別の送り速度を選択し、前記選択された工具の送り速度によって指定された加工プログラムを再実行するものである。すなわち、タスクの異常状態を判定し、異常状態となったタスクが工具の送り速度の影響を受けるタスクか否かを判定し、その判定結果に基づいて工具の送り速度を変えて再実行させるものである。

【0065】したがって、例えば、NC装置の自動加工によってエラーが発生したと判定したときは、工具速度を変えて再度自動加工を行なうことができる。

【0066】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを走行させ、タスクスイッチの発生時に、前記カウンタの値を読み込み、タスク毎に備えた前記カウンタの値と、これから実行が開始されるタスクの前記カウンタの値を記録し、実行権が無くなったタスクの今回の走行時間を計算し、前記タスクの合計走行時間を計算し、前記計算された合計走行時間を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常状態を判定し、前記タスクが工具の送り速度の影響を受けるタスクか否かを判定し、工具の送り速度の影響を受けるタスクであると判定した場合は別の送り速度を選択し、前記選択された工具の送り速度によって指定された加工プログラムを再実行するものである。すなわち、タスクの異常状態を判定し、異常状態となったタスクが工具の送り速度の影響を受けるタスクか否かを判定し、その判定結果に基づいて工具の送り速度を変えて再実行させるものである。

【0067】したがって、例えば、NC装置の自動加工によってエラーが発生したと判定したときは、別の加工プログラムで、再度自動加工を行なうことができる。

【0068】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェ

アを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間を前記カウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを中断させてタスクの走行時間を制限するものである。すなわち、タスクの最大走行時間を制限する機能を有するものである。

【0069】したがって、例えば、NC装置機能を実現するタスクの、走行時間による制限を行なう。すなわち、各タスクに最大の走行時間制限を行うことができる。

【0070】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムにおいて、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間をカウンタに書き込み、前記カウンタに書き込まれた経過時間の経過後に前記タスクを停止させ、前記制御ソフトウェアを複数ブロック解析することにより、割り当てられた時間の範囲内で、複数のブロックの処理を行なうものである。すなわち、タスクの最大走行時間を制限し、各タスクに割り当てられた範囲内で、可能な限り多くのブロックの処理を行なわせるものである。

【0071】したがって、各タスクに予め指定された時間内で最大の処理を行なうので、各タスクに無駄時間がなくなる。例えば、NC装置の機能を、分担して実現しているタスクの制御をきめ細かく実行し、各タスクの処理時間を最適化する。また、NC装置の制御ソフトウェア実行システムの異常を、システム自体が判定することができる。

【0072】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記計算された基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを走行させるものである。

【0073】したがって、全てのタスクを周期的に起動できるようになるので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0074】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを所定の割合走行させるものである。

【0075】したがって、全てのタスクの走行時間の割合を指定できるようになるので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0076】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の時間連続して走ることを保証するために、現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記所定時間前記タスクを走行させるものである。

【0077】したがって、全てのタスクの連続走行時間を指定できるようになるので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0078】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の単位時間毎に、予め指定した割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間が前記システムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを走行させるものである。

【0079】したがって、全てのタスクを周期的に起動して、さらに走行時間の割合も指定できるので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上さ

せることができる。

【0080】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク毎に予め指定した所定の時間毎に、予め指定した所定の単位時間は、連続して走ることを保証するために、現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記経過時間の経過後に前記タスクを所定時間連続して走行させるものである。

【0081】したがって、全てのタスクを周期的に起動して、さらに連続した走行時間の指定もできるので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0082】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムの基準クロックを記録し、前記タスクの実行権がなくなったときの前記記録手段のデータから前記タスクが走行した時間を計算し、前記タスクの走行時間の合計を計算し、前記タスクの合計走行時間を記録するものである。

【0083】したがって、全てのタスクの走行時間を記録できるので、各タスクへの走行時間の割当てを最適なものにすることができる、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0084】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムの基準クロックを記録し、前記タスクの実行権がなくなったときの前記システムの基準クロックのデータから前記タスクが走行した時間を計算し、前記タスクの走行時間の合計を計算し、割り込み発生時の割り込み処理も優先順位の高いタスクによって計算し、前記タスクの合計走行時間および割り込み処理の走行時間を記録するものである。

【0085】したがって、割り込み処理の走行時間も記録できるので、割り込み処理の時間を最適なものにでき、各タスクに無駄時間がなくなり、NC装置の処理速

度を向上させることができる。

【0086】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記タスク毎に基準クロック数を記録し、前記経過時間の経過後に前記タスクを走行させ、前記タスクの実行権が移動するたびに前記基準クロック数を加算し、前記経過時間の経過後に前記タスクを所定の割合走行させると共にタスクや割り込み処理の走行時間を計測するものである。

【0087】したがって、全てのタスクの走行時間を記録でき、さらに全てのタスクの走行時間の割合を指定でき、各タスクへの走行時間の割当てを最適なものにすることができる、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0088】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが、前記基準クロック数分経過したことを判断し、運転条件により前記走行割合を変更するものである。

【0089】したがって、NC装置の運転状態により、各タスクへの走行時間の割当てを自動的に最適なものとするので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0090】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスイッチング処理を行なうスケジューラを設け、前記スケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過時間を計算し、前記計算された経過時間がシステムの基準クロックの何倍かを計算し、前記システムの基準クロックが前記基準クロック数分経過したことを判断し、前記タスクスイッチの発生時に、タスク毎に経過クロック数を

読み込み、前記タスク毎に備えた前記クロック数の合計を記録し、前記クロック数経過後に前記タスクを走行させるとともに、運転を実行しながら自動的に各タスクの走行時間の割合を変更するものである。

【0091】したがって、NC装置を運転しながら、各タスクに割り当てる走行時間の割合を変化させて、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0092】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した所定の割合で走ることを保証するために現在からの経過クロック数を計算し、前記クロック数を記録し、前記クロック数の経過後に前記タスクを走行させ、タスクスイッチの発生時に、タスク毎に経過クロック数を読み込み、タスク毎に備えた前記経過クロック数の合計を記録し、割り込み発生時の割り込み処理も優先順位の高いタスクによって処理し、各タスクの予想全走行割合を指定し、前記指定された予想全走行割合と実際の走行時間のずれの許容度を指定し、前記指定された許容度を含めた予想走行割合から予想全走行時間を計算し、前記計算された予想全走行時間と実際の走行時間との比較によってタスクの異常事態を判定し、運転を行なながらタスクの異常事態を判定するものである。

【0093】したがって、走行時間の比較により異常が発生したタスクを特定できるので、NC装置の動作不良原因を追求し易い。

【0094】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスケジューラによるタスクスイッチの制御に基づき、制御ソフトウェアを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過クロック数を計算し、前記クロック数を記録し、前記クロック数の経過後に前記タスクを中断し、前記タスクの走行時間を制限するものである。

【0095】したがって、各タスクに最大の走行時間制限をすることで、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

【0096】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、メモリ内部のリアルタイムオペレーティングシステム内に異なるタスクのスケジューラによるタスクスイッチの制御に基づき、制御ソフトウェ

アを実行する複数タスクの実行システムの制御方法において、前記システムが、タスク単位で予め指定した時間のみ走ることを保証するために現在からの経過クロック数を計算し、前記クロック数の経過後に前記タスクを停止し、前記制御ソフトウェアを複数ブロック解析することにより、割り当てられた時間の範囲内で、複数のブロックの処理を行なうものである。

【0097】したがって、各タスクに予め指定された時間内で最大の処理を行うので、各タスクに無駄時間がなくなり、NC装置の処理速度を向上させることができる。

#### 【0098】

【発明の実施の形態】以下、この発明に係る制御ソフトウェア実行システムを図について説明する。

【0099】【実施例1】まず、実施例1の構成について説明する。図1は、実施例1に係るNC装置の制御ソフトウェア実行システムA1の全体構成を示すブロック図であり、この制御ソフトウェア実行システムA1は、以下の要素より構成されている。1は各部の動作を制御したり、数値制御に必要な演算を実行するメインCPU1、2はシステムの各部を結ぶシステムバス、3はNC装置の主要機能を実現する制御ソフトウェアなどを格納するROM(不揮発性記憶手段)、4はデータの一時記憶やワークエリアなどに用いるRAM(揮発性記憶手段)、5は外部との間でシリアル通信によりデータのやり取りを行なうSIOインターフェース部である。

【0100】また、図1において、6はNC装置の運転状態を表示したり、NC装置に与えた指令を確認するためにデータ表示を行う表示装置、7はNC装置に対して指令を与えるための入力手段としてのキーボード、8はサーボモータを制御するための指令を演算するサーボ制御部であり、該サーボ制御部8はメインCPU1とは別の専用CPUを備えてもよい。

【0101】さらに、図1において、9はサーボ制御部8から受け取った指令を電力増幅してサーボモータや工作機械主軸(図示せず)を駆動するサーボアンプ、10は工作機械(図示せず)の加工部を制御するためのサーボモータ、11は工作機械との間でサーボ制御指令以外のデータをやり取りするためのプログラマブルコントローラ部(以下、PC部という)、12はメインCPU1に入力される、システムクロック(図示せず)と外部割り込み信号を表している。ここで、システムクロックはNC装置全体を制御するために同期をとるためのクロック信号であり、また、外部割り込み信号は電源異常や非常停止などイベント(緊急の出来事)の発生をメインCPU1に通知するための信号である。

【0102】また、図1において、200は内部割り込み信号であり、この信号はメインCPU1により、その発生やタイミングなどが制御される。201はリアルタイムクロックであり、メインCPU1により制御され、

内部割り込み信号 200 を制御したり、時間を読み取ったりする。さらに、202 はリアルタイムクロック 201 の内部にあるカウンタレジスタであり、これは外部からクロック信号が入力されるたびにその値を 1 ずつ減らし、0 になると再び初期値が設定されるか、あるいはメイン C P U 1 から再び値を設定されるまで動作を止める。この動作の選択や初期値の設定、変更はメイン C P U 1 により任意に実行される。なお、203 はリアルタイムクロック 201 に対してクロック信号を出力する水晶発振器である。

【0103】次に、上記の構成を有する N C 装置の制御ソフトウェア実行システム A 1 の動作について説明する。通常、システムクロックとしては 60 Hz から 100 Hz (1 秒間に 60 回から 100 回) を用いる。システムクロックをこの数値以上に速くすると、その割り込み処理を実行する回数が多くなり過ぎ、N C 装置の機能を実行するタスクの走行可能な時間が少なくなってしまうからである。しかし、タスクの細かい制御に対しては、これでは最小単位が大き過ぎるため、割り込みは使用せずに、例えば、汎用のリアルタイムクロック 201 を使用する。水晶発振器 203 によって、リアルタイムクロック 201 に、例えば、10 MHz のクロック信号を入力すると、リアルタイムクロック 201 はその内部カウンタレジスタ 202 を 1 ずつ減らしていく。10 MHz のクロック信号の入力の場合は 0.1 μs (0.000001 秒) 単位で測定することができる。

【0104】リアルタイムクロック 201 は、また、プログラマブルインターバルタイマとしての役割も果たすことができる。これは、指定した初期値が 0 になったときに C P U 1 に対して割り込みを発生させる機能である。通常リアルタイムクロック 201 は、上記のようにどちらの使い方もできるタイマを複数チャンネル持っている。

【0105】各タスクは、予め、あるいは、そのタスクが開始させられるときに S V C (スーパーバイザーコール、または、サービスコール) によってその実行周期を登録する。ここで、S V C とは、タスクが R T O S に対して何らかのサービスを要求するときの総称である。何らかのサービスとは、例えば、何らかの条件が成立するまで自分自身の実行を停止させたり、他のタスクと通信を行なったり、他のタスクを起動させたり (実行を開始させたり)、停止させたりすることである。

【0106】このデータは、R T O S が、例えば、図 2 に示すような “タスク走行周期テーブル” 50 に登録する。図 2 は、タスク毎に “走行周期” と “繰り返し回数” を有するデータ構造である。ここで、走行周期とは、タスクの次に走るまでのクロック数の値であり、また、繰り返し回数とは、何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば  $\infty$  を示すものとし、その場合にあっては、タスクは繰り

返し起動するものとする。このタスク走行周期テーブル 50 を用いて、R T O S は、タスクに起動がかかるたびに図 3 に示すような “タスク走行待ちリスト” 51 を作成する。

【0107】図 3 において、タスク走行待ちリスト 51 の中の各走行待ちブロック 52 は、それぞれ、次の走行待ちブロックの位置を示す “リンク”、 “タスク名”、今現在からどれだけの時間が経過した後に走らなくてはならないかの “走行予定クロック数”、同時に走らせなくてはならないタスク名を示す “同時タスク名”、それに T C B の位置を示す “T C B ポインタ” 等の要素から成るブロックである。

【0108】以下に、図 4 を用いて具体的な動作について説明する。説明の都合上、水晶発振器 203 に対して入力するクロック信号は、10 MHz として内部カウンタレジスタ 202 は、0.1 μs 毎に 1 減るものとする。すなわち、計測の最小単位は 0.1 μs とする。また、説明中、リストを操作している最中など必要な処理中は、割り込みは禁止しているものとする。さらに、走行待ちブロック 52 を作成する場合には、タスク名としては当該タスクを示す名前を入れておくものとする。

【0109】図 4 は、R T O S が、タスク走行待ちリスト 51 を作成するときの手順を示したものである。図において、まず、B 1 では、要求のあった走行周期を内部クロック数に変換する。例えば、要求が 10 m 秒であれば、100,000 (100,000 × 0.1 μs = 10 m 秒) である。これを R T (request time) とする。次に、B 2 では、現在の走行待ちリストに走行待ちブロックがあるか否かを判定し、走行待ちブロックがないと判定した場合には、B 3 に処理が移行し、反対に、走行待ちブロックがあると判定した場合には、B 6 に処理が移行する。

【0110】B 3 は、走行待ちリスト 51 に走行待ちブロック 52 がなかった場合における処理であり、走行待ちブロック 52 を作成し、B 4 では、走行待ちリスト 51 の先頭に走行待ちブロック 52 を置く。その後、B 5 では、走行待ちリスト 51 の先頭ブロックの走行予定クロック数を、リアルタイムクロック 201 のカウンタレジスタ 202 に設定する。

【0111】次に、B 6 は、走行待ちリスト 51 に走行待ちブロック 52 があると判断した場合における処理であり、最終タスク走行時間 L T を 0 にして、タスクインデックス T I を 1 にする。ここで、L T とは当該タスクを走らせるのは現在からどれだけのクロック数かを計算するための変数であり、T I は走行待ちリスト中で当該タスクを示すインデックスである。その後、B 7 では、現在のリアルタイムクロック 201 のカウンタレジスタ 202 の値を読み込んで、先頭の走行待ちブロック 52 の走行予定クロック数に入れる。次に、B 8 において、走行待ちリスト 51 内の、全ての走行待ちブロック 52

を調べ終ったか（たどったか）否かを判定する。

【0112】その結果、B8において、全ての走行待ちブロック52を調べ終っていないと判定した場合には、B9に処理が移行する。B9では、走行待ちブロック51が、走行待ちリスト51中にあったと判断した場合の処理であり、走行待ちリスト51の中から、T1の示す走行待ちブロック52の、走行予定クロック数を取り出してLTに加える。

【0113】その後、B10において、RTとLTとを比較する。B11は、RTとLTが等しい場合（RT=LT）の処理であり、走行待ちブロック52を作成する。このとき、走行予定クロック数は0にしておく。B12は、B11において作成した走行待ちブロック52を、走行待ちリスト51中のT1の示すブロックの同時タスク名へつなぐ。ここで、つなぐとは、例えば、B11において作成したブロックのアドレスをT1が示す走行待ちブロック52の同時タスク名の場所に置くなど、要するにB11において作成したブロックを、直ちに見つけられるような配置に設定することである。

【0114】B13は、RTがLTより大きな場合（RT>LT）の処理であり、T1の値を、1プラスして、B8へ戻り同じ処理を繰り返す。

【0115】B14は、RTがLTより小さな場合（RT<LT）の処理であり、走行待ちブロック52を作成する。このとき、走行予定クロック数は、RT-LT+(T1が示すより1つ前の走行待ちブロック52の走行予定クロック数)の値にする。その後、B15では、T1が示すブロックの走行予定クロック数をLT-RTとし、B16では、T1が示す走行待ちブロック52とその前のブロックとの間にB14において作成した走行待ちブロック52を挿入する。その後、B17では、B16で挿入したブロックが、走行待ちリスト51の先頭か否かを判定し、B18において、先頭であると判定した場合には、走行待ちリスト51の先頭ブロックの走行予定クロック数を、リアルタイムクロック201のカウンタレジスタ202に設定する。

【0116】B19では、B8において、全ての走行待ちブロックを調べ終ったと判定した場合に、走行予定クロック数がRT-LTの走行待ちブロック52を作成し、その後、B20では、B19において作成した走行待ちブロック52を走行待ちリスト51の最後に挿入する。

【0117】図5は、走行待ちリスト51の例である。この例では現在から30クロック後にタスクAを起動して、さらにそれから20クロック後（すなわち、現在から50クロック後）にタスクBを起動することを示している。このとき、タスクCを35クロック後に起動する要求がRTOSに対して発せられたとすると、図4に示したフローチャートの手順にしたがって走行待ちリストは図6に示すようになる。

【0118】図4に示したB5や、B18において走行待ちリスト51の先頭ブロックの走行予定クロック数を、リアルタイムクロック201のカウンタレジスタ202に指定し、その指定したクロック数が経過した後、“タスク起動割り込み”が発生する。このタスク起動割り込みの処理について図7を用いて説明する。

【0119】図7において、C1では、走行待ちリスト51の先頭の走行待ちブロック52を走行待ちリスト51から切り離す処理を実行し、その後、C2では、走行待ちリスト51が空になったか否かを判定する。また、C3は、走行待ちリスト51が空になったと判定した場合の処理であり、リアルタイムクロック201のカウンタレジスタ202に0を入れるなどの手段により、以後のタスク起動割り込みが発生しないように、タスク起動割り込みを禁止する。

【0120】C4は、C2において、走行待ちリスト51に走行待ちブロック52が残っていると判定した場合の処理であり、先頭の走行待ちブロック52の走行予定クロック数を、リアルタイムクロック201のカウンタレジスタ202に設定する。その後、C5では、C1において切り離した走行待ちブロック52から起動するタスクを取り出して、そのタスクのTCBをTCB待ち行列の先頭に配置する。

【0121】その後、C6では、今走らせたタスクの、タスク走行周期テーブル50（図2参照）の周期回数を判定する。C7は、走行回数が∞（負）の場合の処理であり、このタスクの走行待ちブロック52の、走行予定クロック数を、タスク走行周期テーブル50の走行周期に変え、走行待ちブロックを走行待ちリストに追加する。

【0122】C8は、走行回数が正の整数だった場合の処理であり、走行予定回数を1減らす。その後、C9では、当該タスクに対して図4に示した処理により、このタスクの走行待ちブロック52を作成して走行待ちリストに追加する。

【0123】C10は、走行回数が0の場合の処理であり、この場合、何の処理も実行せずに処理を終了する。C7とC9の処理の後、C11では、RTOSのスケジューラへ飛ぶ。スケジューラの処理は、従来におけるスケジューラ処理と同様であるので、その説明を省略する。

【0124】次に、タスク起動割り込み以外の一般的な割り込み処理について、図8を用いて説明する。D1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。ただし、この処理は、通常メインCPU1により割り込みが発生したときに自動的に実行される。D2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。

【0125】その後、D3では、ICBに、自分自身の

割り込み処理の、実行環境を格納保存する。この中には、次に実行する命令のアドレスとしてD12の処理における先頭アドレスをICBに格納することも含まれる。また、D4では、今作成したICBを、スケジューラが実行するICB待ち行列に追加する。その後、D5は、D1の割り込みの禁止を解除する。これ以降は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのものである。

【0126】次に、D6は、割り込まれた処理が、割り込み処理あるいはRTOSか、それともタスクの処理中かを判定する。その結果、D7は、割り込まれた処理が割り込み処理またはRTOSを実行中であると判断した場合であって、このときは割り込まれた処理に直接戻る。反対に、D8は、割り込まれた処理が、タスクの処理を実行中であると判断した場合であって、このときは、まず割り込まれたタスクのTCBを探し出す。その後、D9では、D8において見つけたTCBへ割り込まれたタスクの実行環境を格納保存する。

【0127】さらに、D10では、作成したTCBを、スケジューラが実行するTCB待ち行列に優先順位の順番に追加する。また、D11では、スケジューラへ飛び、スケジューラからD12に処理が戻ってくるが、スケジューラへ飛んだ後すぐにこの処理が実行されるとは限らず、他のICBが処理された後に戻る場合もある。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理が異なる。さらに、D13は、割り込み処理の終了であり、通常のサブルーチンへのリターン命令が一般的であり、このリターン命令によって、スケジューラのICB処理に戻る。

【0128】上記のとおり、実施例1に係るNC装置の制御ソフトウェア実行システムA1は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSにシステムクロックよりも細かい精度でタスク起動の制御が行なえる機能を追加し、NC装置の制御ソフトウェアを実行するものである。

【0129】実施例1によれば、NC装置の制御ソフトウェア実行システムA1は、NC装置機能を実現するタスクの時間による制御を小さな単位で行なえるようになり、さらにタスクを起動する周期を簡単に指定できる。そのため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御できるので、システムの構築が容易になると共にNC装置の処理速度が向上する。

【0130】【実施例2】次に、実施例2の構成について説明する。実施例2に係るNC装置の制御ソフトウェア実行システムA2の全体構成を示す図9の内容は、実施例1において示した図1とほぼ同様である。ただし、実施例1にあっては、リアルタイムクロック201の割り込みポートを1チャンネルしか使用しなかつたが、本実施例においては3チャンネル用いている点が異なる。

【0131】以下、構成を詳細に説明する。この制御ソフトウェア実行システムA2の1~11までは、実施例1の図1に示した構成と同様であるため、その説明を省略する。図9において、200はメインCPU1に入力される、内部割り込み信号である。この信号は、メインCPU1により、その発生およびタイミングを制御することができる。201はリアルタイムクロックであり、メインCPU1により制御され、内部割り込み信号200を制御し、あるいは時間を読み取ったりするものである。

【0132】また、202はリアルタイムクロック201の内部にあるカウンタレジスタである。これは外部からクロック信号が入力されるたびに、その値を1ずつ減らし、0になると再び初期値が設定される。なお、初期値はメインCPU1から自由に設定できる。なお、203はリアルタイムクロック201にクロック信号を入力する水晶発振器である。

【0133】また、204、205はリアルタイムクロック201の内部にある第2、第3のカウンタレジスタである。これは外部からクロック信号が入力されるたびにその値を1ずつ減らしていき、0になると再び初期値が設定されるか、あるいはメインCPU1から再び値を設定されるまで動作を止める。この動作の選択や初期値の設定、変更是メインCPU1により任意に実行される。また、202、204、205はそれぞれ独立に設定、解除ができる。本実施例ではカウンタレジスタ202、204、205をタスク走行時間計測用、基準時間割り込み用、走行時間警告用にそれぞれ用いる。

【0134】次に、NC装置の制御ソフトウェア実行システムA2の動作について説明する。図10は、本実施例において必要なデータ構造を示したもので、210は“タスク開始チェックメモ”であり、RTOSが、タスクを起動する直前に図9に示したリアルタイムクロック201の内部カウンタレジスタ202の値を記録しておき、その後、タスクが終了してRTOSに戻ってきたときの内部カウンタレジスタ202との差を求めるによりタスクが走った時間を求めるものである。

【0135】また、211は“基準周期”を格納しておく領域である。システムに対して予め1つの基準周期を登録しておく。この周期毎に各タスクはどれくらい走行するかを指定する。この指定は時間でもチック数でも構わないが、ここでは説明の都合上、チック数に変換しておくものとする。

【0136】さらに、212は“タスク走行状態メモ”である。各タスクは予め、あるいは、そのタスクが開始させられるときにSVCによって、その実行の割合を登録する。例えば、タスク毎の指定走行周期、すなわち、タスクAが10%，タスクBが5%，タスクCが20%，基準周期が20m秒のとき、20m秒毎にタスクAが2m秒、タスクBが1m秒、タスクCが4m秒走れば

よい。

【0137】ここで、実施例1と同様、図1において、水晶発振器203へ入力するクロック信号は10MHzとして内部カウンタレジスタ202は、0.1μ秒毎に1減少する、すなわち、計測の最小単位は0.1μ秒とするとA、B、C各タスクはクロック数200,000チックの間にそれぞれ20,000,10,000,40,000チック分走ればよいことになる。このチック数をタスク走行状態メモ212の、"走行チック数"の欄に格納する。ここで、走行する割合を指定しないタスクに対しては、この欄を0に設定しておく。

【0138】また、"走行残りチック数"の欄は、1回の基準周期が終了する毎に、走行チック数の欄をコピーし、タスクが走ったチック数だけ減算していく。1回の基準周期が終る毎に、すべてのタスクの、この欄の内容が0になっていればよい。当然走行チック数の合計は基準周期よりも少なくなければならない。そのエラーチェックは、例えば、タスク状態メモに新たに登録する毎に行なえばよい。ただし、以下の説明においては、このエラーチェックについては省略する。

【0139】次に、"繰り返し回数"の欄に設定するのは、何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば∞を示すとする。その場合には、当該タスクは繰り返し起動するものとする。

【0140】また、"走行禁止フラグ"の欄は、スケジューラが用いる。このフラグがオンになっているタスクはスケジューリングの対象から外す。すなわち、このフラグがオンのタスクはプライオリティが高くても走ることができない。

【0141】次に、割り込み処理の動作について説明する。制御ソフトウェア実行システムA2は、稼働し始めたときなど走行割合を保証するタスクが動作し始める前に、予めリアルタイムクロック（プログラマブルインターバルタイマ）201（図9参照）の内部カウンタレジスタ204に対して、基準周期211（図10参照）の値を設定しておく。

【0142】図11は、基準周期割り込み処理の手順について説明したフローチャートであり、E1では、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常、この処理はメインCPU1によって自動的に実行される。E2では、タスク走行状態メモ212（図10参照）の走行残りチック数の合計を求める。E3では、その合計が0以上か否かを判定し、0以上ではないと判定した場合には、E4においてエラー処理を行なうが、通常は発生しないのでここでは、その説明を省略する。

【0143】反対に、E3において、合計が0以上であると判定した場合には、E5においてタスク走行状態メモ212（図10参照）の走行チック数の欄の値を走行

残りチック数の欄に、タスク毎にコピーする。その後、E6では、再びリアルタイムクロック（プログラマブルインターバルタイマ）201（図9参照）の内部カウンタレジスタ204に対して、基準周期211（図10参照）の値を設定する。

【0144】その後、E7では、タスク走行状態メモ212（図10参照）のすべてのタスクの走行禁止フラグを、オフにする処理を行う。この処理により、再びすべてのタスクがスケジューリング対象になる。さらに、E10では、基準周期211（図10参照）からタスク走行状態メモ212の走行残りチック数の合計を減算した値をリアルタイムクロック201（図9参照）の走行時間警告用の内部カウンタレジスタ205に設定する。この結果、周期を保証したタスクだけを走らせる時間しかなくなつたときに走行時間警告の割り込みが発生する。

【0145】その後、E9では、割り込まれた状態がタスク処理中か否かを判定し、タスク処理中であると判定した場合には、E10において、タスクの実行環境をTCBへ保存し、E11では、E10において保存したTCBを、TCB待ち行列につなぎ、さらに、E12では、割り込みを許可し、E13では、スケジューラへ飛ぶ。

【0146】反対に、E9において、タスク処理中ではないと判定した場合には、E14において、割り込まれた処理に戻る。このとき、割り込みの禁止／許可は割り込まれたときの状態に戻る。

【0147】以上の説明においては、以後の割り込み説明において、従来例のように、割り込み処理自身のアドレスをリストに登録しておいて、多重割り込み（割り込み処理中の他の割り込み）を許可する処理を実行していないが、その場合にあっても、従来例のように、多重割り込みを許す処理を実行してもよい。ただし、その分オーバヘッド（必要な処理が行なわれるまでに要する無駄時間）が発生するので、タスク走行割合などを余分に設定しておく必要がある。

【0148】図12は、走行時間警告割り込み処理の手順について説明したフローチャートである。F1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常、この処理はメインCPU1によって自動的に実行される。F2は、タスク走行状態メモ212（図10参照）の走行残りチック数が、0または負の全てのタスクの走行禁止フラグをオンにする。

【0149】F3では、全ての走行禁止フラグがオフであるTCBが、全ての行禁止フラグがオンのTCBの前に来るよう、タスク処理の待ち行列を作り直す。すなわち、走行禁止フラグがオンのタスクの方がフラグがオフのタスクよりも前に走るようにする。F4は、割り込まれた状態がタスク処理中ならば、タスクの実行環境をTCBへ保存して、割り込みを許可し、スケジューラへ

飛ぶ。反対に、割り込まれた状態がタスク処理中でなければ、割り込まれた処理に復帰する処理である。

【0150】図13は、基準周期割り込み以外の、一般的な割り込み処理の手順を示すフローチャートである。G1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理はメインCPU1によって自動的に実行される。G2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。G3は、ICBに、この割り込み処理（自分自身の）の、実行環境を格納（保存）する。この中には、次に実行する命令のアドレスとして後述するG15の処理の先頭アドレスをICBに格納することも含まれる。

【0151】G4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。G5は、G1の割り込みの禁止を解除する。これ以降は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのものである。

【0152】G6は、割り込まれた処理が、割り込み処理か、あるいは、RTOSか、それともタスクの処理中かを判定する。G7は、割り込まれた処理が割り込み処理またはRTOSを実行中の場合で、このときは割り込まれた処理に直接戻る。G8は、割り込まれた処理がタスクの処理を実行中のときで、このときにあっては、まず、割り込まれたタスクのTCBを探し出す。G9は、G8において見つけたTCBへ割り込まれたタスクの実行環境を格納する。

【0153】G10は、現在処理を行なっているタスクが、走行する割合を保証する対象のタスクであるか否かを判定する処理である。具体的には、タスク走行状態メモ212（図10参照）の走行残りチック数の欄が1以上のタスクか否かを判定する。G11は、G10において対象と判定した場合における処理であり、内部カウンタレジスタ202（図9参照）の値を、基準周期211（図10参照）に保存してある値から減算する。この結果が当該タスクの、この回に連続して走ったチック数である。これをタスク走行状態メモ211の当該タスクの走行残りチック数の欄から減算する。

【0154】G12は、G11において減算した値を、リアルタイムクロック201（図9参照）の走行時間警告用の内部カウンタレジスタ205の値に加える。走行周期を保証したタスクが、走行し終った分だけ、走行周期を保証したタスク以外のタスクを走らせるための時間が増えるわけである。

【0155】G13は、作成したTCBを、スケジューラが実行するTCB待ち行列に追加する。このとき、タスク走行状態メモ212（図10参照）の走行禁止フラグの欄が1つのタスクでもオンになっていれば、TCB待ち行列において、走行禁止フラグがオフのタスクのTCBは、走行禁止フラグがオンのタスクのTCBよりも

前につなぐ。

【0156】G14は、スケジューラへ飛ぶ。G15は、スケジューラから戻ってきたときの処理であり、スケジューラへ飛んだ後すぐにこの処理が実行されることは限らず、他のICBが処理された後の場合もあるが、いずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理内容が異なる。G16は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的であり、このリターン命令によって、スケジューラのICB処理部に戻る。

【0157】次に、図14を用いて、RTOSのスケジューラの動作について説明する。以下、説明の都合上、本実施例においては、優先順位が最低であって、外部に対して何も処理を行なわない、アイドルタスクが常に走っているものとして説明を行なう。

【0158】H1は、割り込み処理（ICB）の待ち行列があるか否かを調べる処理である。H2は、H1において、割り込み処理の待ち行列があったと判断した場合の処理であり、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行おうとするリストの整合性が保たれなくなるためである。

【0159】H3は、割り込み処理の待ち行列のリストから先頭のICBを取り除く。H4は、割り込み禁止を解除する。H5は、H3において取り除いたICBからG3（図13参照）において格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して、それを実行する。これが、G15（図13参照）の処理である。H6は、G16（図13参照）から戻ってくるところであり、再び割り込みを禁止する。H7は、H3において取り除いたICBをICBプールへ返す。H8は、割り込み禁止を解除してからH1へ戻る。後は割り込み処理待ち行列がある間はH1からH7の処理を繰り返す。

【0160】H9は、H1において、割り込み処理の待ち行列がなかったときの処理であり、ここでは、TCBの待ち行列があるか否かを判定する。H10は、H9において、TCBの待ち行列があると判定した場合の処理であり、割り込みを禁止し、その待ち行列を実行するために最初に割り込みを禁止する。H11は、当該TCBのタスクが走行割合を保証するタスクである場合には、タスク開始チックメモ210（図10参照）にリアルタイムクロック201（図9参照）の走行時間計測用の内部カウンタレジスタ202（図9参照）の値を保存しておく。

【0161】H12は、割り込み禁止を解除する。H13は、H11でタスク処理の待ち行列（リスト）から取り除いたTCBからG9（図13参照）において格納し

たタスクの実行環境を取り出して、それを実行する。

【0162】以上で述べたように、実施例2に係るNC装置の制御ソフトウェア実行システムA2は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSに、タスク走行時間の保証を与える制御を実行できる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0163】実施例2によれば、NC装置の制御ソフトウェア実行システムA2は、NC装置機能を実現するタスクの走行時間による制御を実行することができる。そのため、タスクに割り当てた時間の無駄を少なくできる。さらに、タスクをその走行時間において制御できるので、システムの構築が容易になる。

【0164】【実施例3】次に、実施例3の構成について説明する。実施例3に係るNC装置の制御ソフトウェア実行システムA3の全体構成を示す図15の内容にあっては、実施例2において示した図9の内容とほぼ同様である。ただし、実施例3にあっては、カウンタレジスタ202と同様にリアルタイムクロック201の内部にあるカウンタレジスタ206が設けられている。実施例3では、このカウンタレジスタ206を連続走行終了割り込み用に用いる。

【0165】次に、実施例3の動作について説明する。図16、図17は、タスクの連続走行を保証するのに必要なデータ構造を示したもので、図16は、“タスク終了チック数”のリスト構造である。220は、“タスク終了チックブロック”を表している。タスク終了ブロックの構成要素は、タスク終了ブロックを連結するための“リンク（ポインタ）”，“タスク名”，“終了チック数”などである。今までの説明と同様本実施例の説明でも内部的には全てチック数により時間の制御を行なうものとする。

【0166】図17は、“タスク連続走行保証テーブル”221の内容を示している。このテーブルには“タスク名”と“連続走行保証チック数”と“実行中フラグ”の欄があり、連続走行保証チック数の値は、システム起動時、またはSVCにより設定する。連続走行を指定しないタスクに対してはこの欄は0に設定しておく。実行中フラグ欄は現在連続走行を保証されたタスクが走っているときはオン、それ以外のときはオフとし、このフラグは同時に、2つ以上オンすることはない。このテーブルはタスク起動のSVCにおいて、そのパラメータとして指定することにすれば必要はなくなるが、以下の説明の都合上、ここではこのテーブルを用いることにする。

【0167】RTOSのスケジューラは、連続走行が保証されているタスクが走行しているときに、割り込みが発生しても割り込み処理の終了後における再スケジューリングは行なわないことによって、タスクの連続走行を保証する。連続走行が保証されているタスクの実行を終

了させるのは“連続走行終了割り込み”が発生したときである。

【0168】RTOSは、タスク起動のSVCが発行されたときは、まず、そのタスクが連続走行を保証されたタスクか否かを判定し、もし該当タスクであると判定した場合には、タスク連続走行保証テーブル（図17参照）から連続走行チック数をリアルタイムクロック201（図15参照）の内部カウンタレジスタ206に設定する。設定したクロック数が経過すると、“連続走行終了割り込み”が発生する。

【0169】最初に、割り込み処理について説明する。図18は、連続走行割り込みの処理手順について示したフローチャートである。I1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常、この処理はメインCPU1によって自動的に実行される。I2は、現在走っている、連続走行を保証されたタスクのTCBを探し出して、現在のタスクの実行環境をTCBへ保存する。

【0170】次に、I3は、当該タスクの実行中フラグ（図17参照）をオフする。I4は、タスク処理の待ち行列を優先順位に応じて作り直す。さらに、I5は、割り込まれた状態がタスク処理中ならばタスクの実行環境をTCBへ保存してスケジューラへ飛び、そうでなければ割り込まれた処理に復帰させる処理である。このとき、復帰すると同時に割り込みを許可する。

【0171】次に、一般的の割り込みの処理手順について図19のフローチャートを用いて説明する。J1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常、この処理はメインCPU1によって自動的に実行される。J2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。J3は、ICBに、この割り込み処理（自分自身）の、実行環境を格納する。この中には、次に実行する命令のアドレスとしてJ13の処理の先頭アドレスをICBに格納することも含まれる。

【0172】J4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。J5は、J1の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。以上の処理は、割り込み禁止時間をできる限り短くするためのものである。J6は、割り込まれた処理が割り込み処理またはRTOSか、それともタスクの処理中かを判定する。

【0173】J7は、割り込まれた処理が割り込み処理またはRTOSを実行中のときで、このときは割り込まれた処理に直接戻る。反対に、J8は、割り込まれた処理がタスクの処理を実行中のときで、このときは、まず、割り込まれたタスクのTCBを探し出す。J9は、現在処理を行なっているタスクが、連続走行を保証する対象のタスクであるか否かを判定する処理である。具体

的には、タスク連続走行保証テーブル（図17参照）の連続走行保証チック数の欄が1以上のタスクか否かを判定する。連続走行保証の対象タスクであると判定した場合には、以下のJ10、J11の処理は実行しないで、J12へ処理が移行する。

【0174】J10は、連続走行保証の対象外のタスクであると判定した場合の処理で、J8において見つけたTCBへ割り込まれたタスクの実行環境を格納する。J11は、TCB待ち行列（図5参照）を優先順位に応じて作り直す。J12は、スケジューラへ飛ぶ。J13は、スケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後、すぐにこの処理が実行されると限らず、他のICBが処理された後の場合もあるが、いずれはここへ戻ってくる。ここからが割込みに固有の本来の割り込み処理であり、割り込み要因によって、その処理内容は異なる。J14は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が多い、このリターン命令によって、スケジューラのICB処理部に戻る。

【0175】以上で述べたように、実施例3に係るNC装置の制御ソフトウェア実行システムA3は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSに、タスクの連続した走行時間の保証を与える制御を実行できる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0176】実施例3によれば、NC装置の制御ソフトウェア実行システムA3は、NC装置機能を実現するタスクに連続した走行時間を保証できるようになる。そのため、タスクに割り当てた時間の無駄を少なくできる。さらに、タスクをその走行時間において制御できるので、システムの構築が容易に行える。

【0177】【実施例4】次に、実施例4の構成について説明する。実施例4に係るNC装置の制御ソフトウェア実行システムA4の全体構成を示す図20の内容は、実施例1において示した図1の内容とほぼ同様である。ただし、実施例1にあっては、リアルタイムクロック201の割り込みポートを1チャンネルしか使用していないが、本実施例では4チャンネル使用している点が異なる。以下、図2～図4および図10の内容を参照して本実施例を説明する。

【0178】すなわち、202はリアルタイムクロック201の内部にあるカウンタレジスタであり、これは外部からクロック信号が入力されるたびにその値を1ずつ減らしていく。0になると再び初期値が設定されるか、あるいはメインCPU1から再び値を設定されるまで動作を止める。この動作の選択や初期値の設定、変更ははメインCPU1から自由に実行することができる。

【0179】また、204、205、206もリアルタイムクロック201の内部にあるカウンタレジスタであ

る。これらも外部からクロック信号が入力されるたびにその値を1ずつ減らしていく。0になると再び初期値が設定される。また、同様に初期値はメインCPU1から自由に設定できる。また、202、204、205、206は、それぞれ独立に設定、解除することができる。本実施例では、カウンタレジスタ202～206をそれぞれタスク走行時間計測用、基準時間割り込み用、走行時間警告用、タスク走行割り込み用に用いる。

【0180】次に、NC装置の制御ソフトウェア実行システムA4の動作について説明する。実施例1と同様に、各タスクは、予め、あるいは、そのタスクが開始させられるときにSVCによってその実行周期を登録する。

【0181】このデータは、RTOSが、例えば、図2に示したような“タスク走行周期テーブル”50に登録する。図2は、タスク毎に“走行周期”と“繰り返し回数”を持ったデータ構造である。走行周期は、タスクの次に走るまでのクロック数の値であり、繰り返し回数は何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば∞を示すとする。その場合は、そのタスクは繰り返し起動するものとする。このタスク走行周期テーブル50を用いて、RTOSは、タスクに起動がかかるたびに、図3に示したような“タスク走行待ちリスト”51を作成する。

【0182】図3において、リストの中における走行待ちブロック52は、それぞれ“タスク名”と、現在からどれだけの時間の経過後に走らなくてはならないかの“走行予定クロック数”，それに同時に走らせなくてはならないタスク名を示す“同時タスク名”から成るブロックである。このタスク走行待ちリストを、RTOSが作成する方法は、実施例1において図4に示したものと同様なので、その説明は省略する。

【0183】また、実施例2において示したように、図10は、タスクの走行割合を保証するのに、必要なデータ構造を示したもので、210は“タスク開始チックメモ”である。RTOSが、タスクを起動する直前に、リアルタイムクロック201（図20参照）の内部カウンタレジスタ202（図20参照）の値を記録しておく。その後、そのタスクが終了してRTOSに戻ったときににおける内部カウンタレジスタ202（図20参照）との差を求ることによって、タスクが走った時間を求める。

【0184】また、図10に示した“基準周期”211，“タスク走行状態メモ”212，“走行残りチック数”の欄，“繰り返し回数”の欄および“走行禁止フラグ”の欄については、実施例2と同様なので、その説明を省略する。

【0185】図4に示したB4や、B16において走行待ちリストの先頭ブロックの走行予定クロック数を、リアルタイムクロック201（図20参照）のカウンタレ

ジスタ202（図20参照）に入れ、その指定したクロック数が経過した後，“タスク起動割り込み”が発生する。このタスク起動割り込みの処理については、実施例1の図7に基づいて説明したものと同様であるので、その説明を省略する。

【0186】図20に示した制御ソフトウェア実行システムA4は、稼働し始めたときなど走行割合を保証するタスクが動き始める前に、予めプログラマブルインターバルタイマ201の内部におけるカウンタレジスタ204に対して、基準周期211（図10参照）の値を設定しておく。この設定したチック数が経過した後に、基準周期割り込みが発生する。

【0187】基準周期割り込みの処理手順については、実施例2の図11に基づいて説明した手順と同様なので、その説明を省略する。また、走行時間警告割り込みの処理手順についても、実施例2の図12に基づいて説明した手順と同様なので、その説明を省略する。また、基準周期割り込み以外の、一般的な割り込みの処理手順についても、実施例2の図13に基づいて説明した手順と同様なので、その説明を省略する。さらに、RTOSのスケジューラの動作についても、実施例2の図14に基づいて説明した手順と同様なので、その説明を省略する。

【0188】ここで、タスク起動割り込み、基準時間割り込み、走行時間警告割り込み、および、一般的な割り込みの各処理について、TCB待ち行列への操作を中心まとめる。

【0189】タスク起動割り込みは、この割り込みによって起動するタスクのTCBを、TCB待ち行列の先頭に挿入する。また、基準時間割り込みは、全てのタスクを走行可能とする。走行時間警告割り込みは、走行時間の割合を保証しているタスクのTCBを、そうでないタスクのTCBよりも前に持ってくると共に、タスク走行状態メモ212（図10参照）の走行禁止フラグをオンすることによって、走行時間警告割り込みが発生したことを、一般的な割り込みに対して知らせる。さらに、一般的な割り込みは、走行時間警告割り込みの処理によって、走行時間警告割り込みが発生した場合には、基準時間割り込みが発生するまでは、走行時間を保証しているタスクのTCBをそうでないタスクのTCBよりもTCB待ち行列の前につなぐ。

【0190】ここで、タスク起動割り込みと一般的な割り込みの優先順位に関して、どちらかを選択しておく。タスク起動割り込みを、優先すればタスクの起動時間は正確になるが、タスクの走行割合は多少不正確になる。反対に、一般的な割り込みを優先すれば、タスクの走行割合は正確になるが、タスクの起動時間は多少不正確になる。これらは、何らかの手段によって閾値を決めておいてよい。例えば、タスクの起動時間の遅れが、ある時間までは走行割合を優先するが、それ以上はタスク起動

を優先するなどの方法を取ればよい。

【0191】以上で述べたように、実施例4に係るNC装置の制御ソフトウェア実行システムA4は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSに、タスク起動周期とタスク走行割合の保証を与えるの制御を行なえる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0192】実施例4によれば、NC装置の制御ソフトウェア実行システムA4は、NC装置機能を実現するタスクの正確な起動時間と走行時間による制御を行なえるようになる。そのため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間で制御できるので、システムの構築が容易に行なえる。

【0193】【実施例5】次に、実施例5について説明する。実施例5に係るNC装置の制御ソフトウェア実行システムA5の全体構成は、実施例1に示したものと同様であるので、その説明を省略する。

【0194】次に、実施例5に係るNC装置の制御ソフトウェア実行システムA5の動作について説明する。実施例1と同様に、各タスクは、予め、あるいは、そのタスクが開始させられるときにSVCによってその実行周期を登録する。

【0195】このデータは、RTOSが、例えば、図2に示したような“タスク走行周期テーブル”50に登録する。このタスク走行周期テーブル50における“走行周期”と“繰り返し回数”については、実施例1と同様なので、その説明を省略する。

【0196】図3において、リストの中の各走行待ちブロックは、それぞれ“タスク名”と、現在からどれだけの時間の経過後に走らなくてはならないかの“走行予定クロック数”，それに同時に走らせなくてはならないタスク名を示す“同時タスク名”からなるブロックである。このタスク走行待ちリストを、RTOSが作成する方法は、実施例1において図4で示したものと同様なので、その説明は省略する。

【0197】また、実施例3と同様に、図16、図17は、タスクの連続走行の走行割合を保証するのに、必要なデータ構造を示したもので、図16は、“タスク終了チック数”のリスト構造である。220は、“タスク終了チックブロック”を表している。タスク終了ブロックの構成要素はタスク終了ブロックを連結するためのポインタ，“タスク名”，“終了チック数”などである。今までの説明と同様本実施例の説明でも内部的には全てチック数で時間の制御を行なうものとする。“実行中フラグ”欄は現在連続走行を保証されたタスクが走っている時はオン、それ以外の時はオフとする。このフラグは同時に2つ以上オンすることはない。

【0198】図17は、“タスク連続走行保証テーブル”221である。このテーブルには“タスク名”と

“連続走行保証チック数”の欄があり、連続走行保証チック数の値はシステム起動時、またはSVCにより設定する。連続走行を指定しないタスクに対してはこの欄は0にしておく。このテーブルはタスク起動のSVCにおいて、そのパラメータとして指定することにすれば、必要はなくなるが、以下の説明の都合上、ここではこのテーブルを用いることにする。

【0199】RTOSのスケジューラは、連続走行が保証されているタスクが走行しているときに、割り込みが発生しても割り込み処理の終了後における再スケジューリングは行なわないことによって、タスクの連続走行を保証する。連続走行が保証されているタスクの実行を終了させるのは“連続走行終了割り込み”が発生したときである。

【0200】RTOSはタスク起動のSVCが、発行されたときには、まずそのタスクが連続走行を保証されたタスクか否かを判定して、もし該当タスクであると判定した場合には、タスク連続走行保証テーブル221(図17参照)から連続走行チック数をリアルタイムクロック201(図1参照)の内部カウンタレジスタ206に設定する。設定したクロック数が経過すると、“連続走行終了割り込み”が発生する。

【0201】連続走行終了割り込みの処理手順については、実施例3に示した図18の内容と同様なので、その説明を省略する。基準周期割り込み以外の、一般的な割り込みの処理手順についても、実施例3に示した図19の内容と同様なので、その説明を省略する。スケジューラについても、従来例と同じであるので、その説明を省略する。

【0202】ここで、連続走行終了割り込みと、一般的な割り込みの処理について、TCB待ち行列への操作を中心まとめる。すなわち、

- ① タスク起動割り込みは、この割り込みによって起動するタスクのTCBを、TCB待ち行列の先頭に挿入する、
- ② 連続走行終了割り込みは、実行中のタスクの実行中フラグをオフして他のタスクの走行を許可する、
- ③ 一般的な割り込みは、連続走行を保証するタスクの場合には、TCB待ち行列のディスパッチを実行しない、がある。

【0203】ここで、上記①と③の優先順位はどちらかを選択しておく。①を優先すればタスクの起動時間は正確になるが、タスクの連続走行時間は多少不正確になる。一方③を優先すれば、タスクの連続走行時間は正確になるが、タスクの起動時間は多少不正確になる。これらは、何らかの手段によって閾値を決めておいてよい。例えば、タスクの起動時間の遅れが、ある時間までは連続走行を優先するが、それ以上は、タスク起動を優先するなどの方法を取ればよい。

【0204】以上述べたように、実施例5に係るNC装

置の制御ソフトウェア実行システムA5は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSに、タスク起動周期とタスクの連続走行の保証を与える制御を行なえる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0205】実施例5によれば、NC装置の制御ソフトウェア実行システムA5は、NC装置機能を実現するタスクの正確な起動時間と連続走行時間による制御を実行することができる。そのため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間内において制御できるので、システムの構築が容易に行なえる。

【0206】【実施例6】次に、実施例6について説明する。図21は、実施例6に係るNC装置の制御ソフトウェア実行システムA6の全体構成を示し、その内容は、実施例1とほぼ同様である。ただし、実施例1にあっては、リアルタイムクロック201の割り込みポートを1チャンネルしか使用しないのに対し、本実施例にあっては、2チャンネル使用している。

【0207】図において、202, 204は、リアルタイムクロック201の内部にあるカウンタレジスタである。カウンタレジスタ202, 204は、それぞれ独立に設定、解除ができる。本実施例にあっては、カウンタレジスタ202, 204をそれぞれタスク走行時間計測用に用いる。

【0208】次に、NC装置の制御ソフトウェア実行システムA6の動作について説明する。本実施例に係る制御ソフトウェア実行システムA6のRTOSは、予め、例えば、システムを稼働したとき等に、リアルタイムクロック201の内部カウンタレジスタ202, 204は、0になると再び初期値が設定され、また、0になつても、割り込みが発生しないように設定しておく。

【0209】また、タスクにはいる直前に、そのカウンタ値を読み取って記憶しておく。また、そのタスクを抜けた直後に、再びカウンタ値を読み取って記憶しておいた値との差を求める。その値が、タスクが走ったクロック数である。その後タスク毎に、この値の和を記憶しておく場所を設けておいて、そこに加えていくことにより、各タスクの走行時間の合計を記録する。

【0210】以下、本実施例に係るNC装置の制御ソフトウェア実行システムA6の動作について、さらに詳しく説明する。図22は、各タスクの走行時間を記録するに必要なデータ構造であり、230はカウンタ値メモ、231はタスク走行時間テーブルであり、図に示す通り、タスク毎の合計チック数の欄がある。図23は、一般的な割り込み処理の処理手順を示すフローチャートであり、図24は、スケジューラの処理手順を示す図である。

【0211】図23は、一般的な割り込みの処理手順につ

いて示したフローチャートであり、K 1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常この処理はメインCPUによって自動的に実行される。K 2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。K 3は、ICBに、この割り込み処理（自分自身）の、実行環境を格納する。この中には次に実行する命令のアドレスとしてK 8の処理の先頭アドレスをICBに格納することも含まれる。K 4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。K 5は、K 1の割り込みの禁止を解除する。これ以降は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためである。

【0212】K 6は、割り込まれた処理が、割り込み処理またはRTOSか、それともタスクの処理中かを判定する。K 7は、割り込まれた処理が割り込み処理またはRTOSを実行中であると判定した場合の処理であって、割り込まれた処理に直接戻る。K 8は、割り込まれた処理がタスクの処理を実行中の場合であって、割り込まれたタスクのTCBを探し出す。K 9は、K 8において、見つけたTCBへ割り込まれたタスクの実行環境を格納する。

【0213】K 10は、リアルタイムクロック201（図21参照）の内部カウンタレジスタ204の値を読み込んで、カウンタ値メモ230（図22参照）から減算する。この結果が当該タスクがこの回に連続して走ったチック数である。K 11は、K 10において求めた値をタスク走行時間テーブル231（図22参照）の当該タスクの合計チック数の欄に加える。K 12は、作成したTCBを、スケジューラが実行するTCB待ち行列に優先順位の順番に追加する。K 13は、スケジューラへ飛ぶ。

【0214】K 14は、スケジューラから戻ってきた後に実行する処理である。スケジューラへ飛んだ後すぐにこの処理が実行されるとは限らず、他のICBが処理された後の場合もあるが、いずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なってくる。K 15は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的的であり、このリターン命令によって、スケジューラのICB処理部へ戻る。

【0215】図24は、本実施例に係るRTOSのスケジューラの動作を説明したものである。以下、説明の都合上、本実施例においては、優先順位が最低で、外部に 対して何ら処理を実行しないアイドルタスクが常に走っているものとする。

【0216】L 1は、割り込み処理（ICB）の待ち行列があるか否かを判定する処理である。L 2は、割り込み処理の待ち行列があったときの処理であり、その待ち

行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加しようとするとリストの整合性が保たれなくなるためである。

【0217】L 3は、割り込み処理の待ち行列のリストから先頭のICBを取り除く。L 4は、割り込み禁止を解除する。L 5は、L 3において取り除いたICBからK 3（図23参照）において格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して実行する。これがK 14（図23参照）の処理である。L 6は、K 15（図23参照）から戻ってくるところであり、再び割り込みを禁止する。L 7は、L 3において取り除いたICBをICBプールへ返す。L 8は、割り込み禁止を解除してからL 1へ戻る。後は割り込み処理待ち行列がある間はL 1からL 7の処理を繰り返す。

【0218】L 9は、TCB待ち行列があるか否かを判断するものであり、L 10は、タスク処理の待ち行列があつたと判定した場合の処理であり、その待ち行列を実行するために最初に割り込みを禁止する。L 11は、割り込み処理の待ち行列のリストから先頭のTCBを取り除く。L 12は、カウンタ値メモ230（図22参照）にリアルタイムクロック201（図21参照）の走行時間計測用の内部カウンタレジスタ204（図22）の値を保存しておく。

【0219】L 13は、割り込み禁止を解除する。L 14は、L 11においてタスク処理の待ち行列（リスト）から取り除いたTCBからK 9（図23参照）において格納したタスクの実行環境を取り出して実行する。このときには、その処理へ飛び、ここへは再び戻ってこない。L 15は、L 9においてTCB待ち行列がないと判断した場合における処理であり、割り込み待ちを行う。

【0220】以上のようにして、任意の時間、本システムを実行させた後、タスク走行時間テーブル231（図22参照）を調べれば、各タスクが実際に走った（処理を行なった）時間を確認することができる。

【0221】なお、タスクの走行時間のみではなく、タスク走行時間テーブル231（図22参照）に走行回数の欄も設けて、タスクの走行回数をも記録するようにしておけば、以下で説明するタスクの走行周期、走行割合、走行時間を決めるときにも役立つ。

【0222】さらに、カウンタ値メモ230（図22参照）だけでなく、タスク終了カウンタ値メモも設けて割り込み処理の先頭で、リアルタイムクロック201（図21参照）の内部カウンタレジスタ204（図21参照）の値を保存しておき、割り込み処理（図23参照）のK 10の部分において、カウンタ値メモ230からタスク終了カウンタ値メモを減算した値をタスクが連續して走行したチック数とすれば割り込み処理の時間も除い

たタスクが純粹に走行した時間が求められる。

【0223】以上で述べたように、実施例6に係るNC装置の制御ソフトウェア実行システムA6は、タスクの実際の走行時間を知ることができる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0224】実施例6によれば、NC装置の制御ソフトウェア実行システムA6は、NC装置機能を実現するタスクの正確な実行時間を知ることにより、各タスクの走行時間の割り当てなどが、容易に行なえるようになり、システムの構築が容易となる。

【0225】【実施例7】次に、実施例7について説明する。実施例7に係るNC装置の制御ソフトウェア実行システムA7の全体構成は、実施例6と同様であるので、その説明を省略する。

【0226】次に、本実施例に係るNC装置の制御ソフトウェア実行システムA7の動作について説明する。本実施例における制御ソフトウェア実行システムA7(図21参照)のRTOSは、予め、例えは、システムを稼働したとき等に、リアルタイムクロック201の内部カウンタレジスタが0になると再び初期値が設定され、また、0になっても、割り込みが発生しないように設定しておく。

【0227】また、タスクにはいる直前に、そのカウンタ値を読み取って記憶しておく。そのタスクを抜けた直後に、再びカウンタ値を読み取って記憶しておいた値との差を求める。その値が、今回このタスクが走ったクロック数である。その後、タスク毎に、この値の和をを記憶しておく場所を設け、そこに加えていくことにより、各タスクの走行時間の合計を記録する。さらに、従来例におけるスケジューラのICBの処理を、スケジューラでなく、優先順位がもっとも高いタスクで実行するものである。

【0228】以下、本実施例に係るNC装置の制御ソフトウェアを実行する動作について、さらに詳しく説明する。図25は、スケジューラの処理手順を示すフローチャートであり、図26は、ICBを処理する最高優先順位を持った、割り込み処理実行タスクの処理手順を示すフローチャートである。データ構造としては、従来例と同じICB(インターフェースコントロールブロック)を用いる。なお、割り込み処理に関しては、従来例と同じであるので、その説明を省略する。

【0229】図25は、本実施例に係るRTOSのスケジューラの動作を説明したものであり、アイドルタスクを設けることにより、常に少なくともアイドルタスクは、実行を行っているものとする。M1は、タスク処理の待ち行列(TCB)があったときの処理であり、その待ち行列を実行するために最初に割り込みを禁止する。

M2は、割り込み処理の待ち行列のリストから先頭のTCBを取り除く。M3は、カウンタ値メモ230(図2

2参照)にリアルタイムクロック201(図21参照)の走行時間計測用の内部カウンタレジスタ204(図21参照)の値を保存しておく。

【0230】M4は、割り込み禁止を解除する。M5は、M3においてタスク処理の待ち行列(リスト)から取り除いたTCBから割り込み処理で格納したタスクの実行環境を取り出して実行する。このときにあっては、その処理へ飛び、ここへは再び戻ってこない。

【0231】図26は、本実施例に係る割り込み処理実行タスクの動作を説明したものであり、N1は、割り込み処理(ICB)の待ち行列があるか否かを判定する処理である。N2は、割り込み処理の待ち行列があったと判断した場合の処理であって、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行なうとするとリストの整合性が保たれなくなるためである。

【0232】また、N3は、割り込み処理の待ち行列のリストから、先頭のICBを獲得する。N4は、割り込み禁止を解除する。N5は、N3で取り除いたICBから、割り込み処理で格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して実行する。N6は、N3で取り除いたICBをICBプールへ返却する。その後は、N1へ戻り、割り込み処理待ち行列がある間はN1からN6の処理を繰り返す。N7は、割り込み処理(ICB)の待ち行列が無くなったときの処理でタスク終了のSVCを発行して他のタスクにメインCPU1の使用権を譲り渡す。

【0233】以上のように、任意の時間、本実施例を走らせた後、タスク走行時間テーブル231(図22参照)を調べれば、各タスクが実際に走った(処理を行なった)時間を知ることができる。

【0234】さらに、割り込み処理もタスクレベルで実行することにより、割り込み処理実行タスクの走行時間が、割り込み処理の時間と考えることができるため、割り込み処理の処理時間も測定できる。また、割り込み処理実行タスク内で、割り込み要因により別の走行時間を記録できるようにしておけば、割り込み要因毎の処理時間測定ができる。

【0235】以上で説明したとおり、実施例7に係るNC装置の制御ソフトウェア実行システムA7は、タスクと割り込み処理の実際の走行時間を知ることができる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0236】実施例7によれば、NC装置の制御ソフトウェア実行システムA7は、NC装置機能を実現するタスクと割り込み処理の正確な実行時間を知ることにより、各タスクの走行時間の割り当てなどが容易に実行できるようになり、システムの構築が容易となる。

【0237】【実施例8】次に、実施例8について説明

する。タスクや割り込み処理の走行時間を計測する手段を備え、さらにタスクの走行時間を変える手段を備えるNC装置において、それを実現するためのNC装置の制御ソフトウェア実行システムA8の全体構成は、タスクの走行時間や割合の指定の仕方によって複数例あるが、この実施例では説明の都合上、タスクの走行割合を指定できる場合について説明する。

【0238】実施例8に係るNC装置の制御ソフトウェア実行システムA8の全体構成は、実施例2(図9参照)とほぼ同様であるので、その説明を省略する。また、タスクの走行割合を保証する処理については、実施例2(図11～図12)とほぼ同様であり、必要なデータ構造も同じであり、タスク開始チェックメモ210(図10参照)、基準周期を格納しておく領域211(図10参照)、タスク走行状態メモ212(図10参照)などを備える。タスク走行状態メモ212(図10参照)は、走行残りチック数、繰り返し回数、走行禁止フラグなどの欄を備える。

【0239】さらに、タスクの走行時間を測定するために、実施例6と同様に、タスク走行時間テーブル231(図22参照)を備える。また、タスク走行時間テーブル231(図22参照)にはタスク毎の合計チック数の欄がある。なお、カウンタ値メモ230(図22参照)はタスク走行状態メモ212(図10参照)と同じ役割を果たすものなので使用しない。また、基準時間割り込み処理の手順、走行時間警告割り込み処理の手順は、実施例2(図11、図12参照)と同様なので、その説明を省略する。

【0240】図27は、一般の割り込みの処理手順について説明したフローチャートであり、O1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理はメインCPU1によって自動的に実行される。O2は、未使用のICBをICBプール(未使用ICBを保持しておく場所)から1つ獲得する。

【0241】O3は、ICBに、この割り込み処理(自分自身)の、実行環境を格納する。この中には、次に実行する命令のアドレスとしてO8の処理の先頭アドレスをICBに格納することも含まれる。O4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。O5は、O1の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのものである。

【0242】O6は、割り込まれた処理が割り込み処理またはRTOSか、タスクの処理中かを判定する。O7は、割り込まれた処理が割り込み処理またはRTOSを実行中であると判断した場合で、このときは割り込まれた処理に直接戻る。O8は、割り込まれた処理がタスクの処理を実行中のときで、このときは、まず、割り込ま

れたタスクのTCBを探し出す。O9は、O8において探し出したTCBへ割り込まれたタスクの実行環境を格納する。

【0243】O10は、内部カウンタレジスタ202(図9参照)の値を、基準周期211(図9参照)に保存してある値から減算する。この結果が当該タスクが今回連続して走ったチック数である。これをタスク走行時間テーブル231(図22参照)の当該タスクの合計チック数の欄に加える。O11は、現在処理を実行しているタスクが、走行する割合を保証する対象のタスクであるか否かを判定する処理である。具体的には、タスク走行状態メモ212(図9参照)の走行残りチック数の欄が1以上のタスクか否かを判定する。

【0244】O12は、O11において対象のタスクであると判定した場合の処理で、O10において求めた値をタスク走行状態メモ211の当該タスクの走行残りチック数の欄から減算する。O13は、O10において求めた値を、リアルタイムクロック201(図9参照)の走行時間警告用の内部カウンタレジスタ205の値に加える。走行周期を保証したタスクが、走行し終った分だけ、走行周期を保証したタスク以外のタスクを走らせるための時間が増えるわけである。

【0245】O14は、作成したTCBを、スケジューラが実行するTCB待ち行列に、優先順位の順番に追加する。ただし、このとき、タスク走行状態メモ212(図10参照)の走行禁止フラグの欄が1つのタスクでもオンになっていれば、TCB待ち行列において、走行禁止フラグがオフのタスクのTCBは、走行禁止フラグがオンのタスクのTCBよりも前につなぐ。O15は、スケジューラへ飛ぶ。

【0246】O16は、スケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後、すぐにこの処理が実行されるとは限らず、他のICBが処理される場合もあるが、いずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。O17は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的であり、このリターン命令によって、スケジューラのICB処理部に戻る。

【0247】図28は、スケジューラの動作を示すフローチャートであり、本実施例においては、優先順位が最低で、外部に対して何も処理を行なわないアイドルタスクが常に走っているものとする。P1は、割り込み処理(ICB)の待ち行列があるか否かを判定する処理である。P2は、割り込み処理の待ち行列があったと判断した場合の処理であり、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を実行しようとするリストの整合性が保たれなくなるためである。

【0248】P3は、割り込み処理の待ち行列のリストから先頭のICBを取り除く。P4は、割り込み禁止を解除する。P5は、P3において取り除いたICBからO3（図27参照）において格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して実行する。これが、O16（図27参照）の処理である。P6は、O17（図27参照）から戻ってくるところであり、再び割り込みを禁止する。P7は、P3において取り除いたICBをICBプールへ返す。P8は、割り込み禁止を解除してからP1へ戻る。後は割り込み処理待ち行列がある間はP1からP7の処理を繰り返す。

【0249】P9は、ICB待ち行列はないと判断した場合の処理であり、タスク処理の待ち行列を実行するために、割り込みを禁止する。P10は、割り込み処理の待ち行列のリストから先頭のTCBを獲得する。P11は、タスク開始チェックメモ210（図10参照）にリアルタイムクロック201（図9参照）の走行時間計測用の内部カウンタレジスタ202（図9参照）の値を保存しておく。P12は、割り込み禁止を解除する。P13は、P10においてタスク処理の待ち行列（リスト）の先頭から、獲得したTCBからO9（図27参照）において格納したタスクの実行環境を取り出して実行する。このときはその処理へ飛んでいって、ここへは再び戻ってこない。

【0250】以上において説明したように、実施例8に係るNC装置の制御ソフトウェア実行システムA8は、タスクの走行割合を設定でき、また、タスクと割り込み処理の実際の走行時間を知ることができる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0251】実施例8によれば、NC装置の制御ソフトウェア実行システムA8は、NC装置機能を実現するタスクと、割り込み処理の正確な実行時間を知ることにより、各タスクの走行時間の割り当てなどが容易に行なえるようになり、システムの構築が容易となる。

【0252】【実施例9】次に、実施例9について説明する。実施例9に係るNC装置の制御ソフトウェア実行システムA9の全体構成は、実施例2（図9参照）とほぼ同様であるので、その説明を省略する。

【0253】次に、NC装置の制御ソフトウェア実行システムA9の動作について説明する。図29は、NC装置の運転モード選択部分の処理手順を示すフローチャートであり、Q1は、NC装置のモードがグラフィックチェックか否かを判定する部分である。Q2は、グラフィックチェックモードであると判断した場合であり、その場合はグラフィック表示タスクの割り当てを増やし、他の、例えば、自動モード処理タスクの割り当てをその分減らす。

【0254】Q3は、グラフィックチェックモードではないと判断した場合で、自動モードか否かを判定する処

理である。Q4は、自動モードであると判断した場合であって、自動モード処理タスクの割り当てを増やし、他の処理タスクの割り当てを減らす。Q5は、手動モードか否かのチェックである。Q6は、手動モードであると判断した場合であって、手動処理タスクの割り当てを増やし、他の処理タスクの割り当てを減らす。それ以後は、その他のモードの場合であるが、その説明は省略する。

【0255】以上において説明したとおり、実施例9に係るNC装置の制御ソフトウェア実行システムA9にあっては、NC装置の運転モードによってタスクの走行割合を変化させて、NC装置の制御ソフトウェアを実行するものである。

【0256】実施例9によれば、NC装置の制御ソフトウェア実行システムA9は、NC装置の運転モードによってタスクの走行割合を変化させることにより、モード毎にタスクに適した時間だけタスクを走らせることができ、NC装置の実行速度が向上する。

【0257】【実施例10】次に、実施例10について説明する。実施例10に係るNC装置の制御ソフトウェア実行システムA10は、同一の加工プログラムを複数回走らせるとき（同じ加工を複数回行なうとき）、初回加工時における各タスクの走行状況に基づいて、2回目以降の加工時における各タスクの走行配分を最適化するものである。

【0258】実施例10に係るNC装置の制御ソフトウェア実行システムA10の全体構成は、実施例2（図9参照）とほぼ同様であるので、その説明を省略するが、以下、説明の都合上、本実施例においては、例えば、実施例8と同様に、タスク毎の走行割合を指定する手段と、実際のタスクの走行状況を記録できる手段とを備えているものとする。

【0259】使用するデータ構造としては、実施例8と同様、タスク開始チェックメモ210（図10参照）、基準周期を格納しておく領域211（図10参照）、タスク走行状態メモ212（図10参照）、タスク走行時間テーブル231（図22参照）などを備える。タスク走行状態メモ212（図10参照）は、走行残りチック数、繰り返し回数、走行禁止フラグなどの欄があり、タスク走行時間テーブル231（図22参照）にはタスク毎の合計チック数の欄がある。

【0260】基準時間割り込みの処理手順、走行時間警告割り込みの処理手順は、実施例2（図11、図12参照）と同様であり、一般的の割り込みの処理手順は、実施例8（図27参照）と同様なので、その説明を省略する。

【0261】図30は、複数回加工時における処理の時間を短縮するための、各タスクの走行割り当て時間を最適化するための手順を示すフローチャートであり、R1は、同一加工の1回目か、2回目以降かを判定する処

である。R 2 は、1 回目の加工時の処理であると判断した場合で、各タスクの総走行時間（チック数）を計測、記録する。R 3 は、走行したタスクの内、アイドルタスクなどの必要のないタスクを除いた、他の必要なタスクの走行時間の合計を計算する。

【0262】R 4 は、R 3 において求めた合計に基づいて、必要なタスクの走行割合を計算する。走る必要がないタスクの走行割合は 0 とする。R 5 は、R 4 において計算した割合をタスク走行状態メモ 212（図 10 参照）の走行割合の欄に登録する。R 6 は、2 回目以降の加工時における処理であり、R 5 で登録したタスク走行状態メモ 212（図 10 参照）の各タスクの走行割合により、各タスクを走行させる。

【0263】以上、説明したとおり、実施例 10 に係る NC 装置の制御ソフトウェア実行システム A 10 は、1 回目の加工によって、最適なタスクの走行割合を自動的に設定する機能を備え、NC 装置の制御ソフトウェアを実行するものである。

【0264】実施例 10 によれば、NC 装置の制御ソフトウェア実行システム A 10 は、機能を実現する各タスクへの走行時間の割り当てを、自動的に最適なものにすることにより、繰り返し加工における加工時間の短縮を実行できる。

【0265】【実施例 11】次に、実施例 11 について説明する。実施例 11 に係る NC 装置の制御ソフトウェア実行システム A 11 は、予め登録された、各タスクの標準の走行割合や、全走行時間と、実際に走っている NC 装置の制御ソフトウェア実行システムの各タスクのデータが、予め指定した割合以上にずれている場合には、タスクの異常と判定する。

【0266】実施例 11 に係る NC 装置の制御ソフトウェア実行システム A 11 の全体構成は、実施例 8（図 9 参照）と同様であるので、その説明を省略する。また、実際に走っている NC 装置の制御ソフトウェア実行システムの各タスクのデータを測定するのは、実施例 8 の場合と同じであるので、その説明を省略する。

【0267】各タスクが正常に走っているか否かを判定するのに必要なデータ構造を、図 31 に示す。250 は、“タスク走行標準データ”で、タスク毎に“走行割合”や“全走行チック数”などの欄がある。これらのデータは、システム生成時に予め作成しておくか、あるいは何らかの SVC によって、そのデータを使うよりも前の、任意の時期に設定しておいてもよい。

【0268】図において、251 は、この割合までは、タスク走行時間のずれがあっても、タスク異常と見なさない許容度を格納する“走行時間判定許容度”であり、また、252 は、この範囲内であれば、タスク走行割合に、ずれがあってもタスク異常と見なさない許容度を格納する“走行割合判定許容度”である。この走行割合判定許容度には下限欄と上限欄があり、ずれがこの範囲内

ならばタスク異常と見なさない。

【0269】各タスクが正常に走っているか否かを判定するための、タスク走行時間テーブル 231（図 22 参照）とタスク標準データ 250（図 31 参照）との比較は、例えば、タスクが切り替わるときに、スケジューラにより行ってもよいし、あるいは定期的に走るタスク異常判定タスクを設けて、そのタスクにより判定を行なつてもよい。

【0270】以下の説明では、タスクが正常に走っているか否かの判定を、異常判定タスクに基づいて行うものとする。また、判定許容度 251 は全走行時間と、走行割合について別々に設けてもよいが、ここでは説明の都合上、同じ値を用いるものとする。

【0271】図 32 は、異常判定タスクの動作を示すフローチャートである。S 1 は、異常判定タスクの先頭で、タスク走行時間テーブル 231（図 22 参照）の、タスク毎の合計チック数の欄の値、全ての和を求める処理である。S 2 は、全てのタスクに対して、以下の S 3 から S 5 までの処理を行なったか否かを判定し、全タスクに対して処理を終了したと判定した場合には、S 6 へ飛ぶ。

【0272】S 3 は、タスク走行時間テーブル 231（図 22 参照）の、当該タスクの合計チック数の値と、標準データ 250（図 31 参照）の全走行チック数を比較する。当該タスクのタスク走行時間テーブル 231（図 22 参照）の合計チック数の値が、標準データ 250（図 31 参照）の全走行チック数の値に走行時間判定許容度 251（図 31 参照）の値を乗じたものよりも小さいか、等しければ、このタスクは正常であると見なす。

【0273】S 4 は、S 3 において、タスクに異常が発生したと判定した場合における処理であり、例えば、オペレータにタスク異常が発生したことを知らせるメッセージを送るなどの、何らかのエラー処理を実行する。S 5 は、タスクの走行割合のチェックである。タスク走行時間テーブル 231（図 22 参照）の合計チック数の値を、全タスク分加えて、それで当該タスクの合計チックを割って、当該タスクの走行割合を求める。その値が標準データ 250（図 31 参照）の走行割合判定許容度の下限と上限との間にあれば、このタスクは正常であると見なす。もし、異常があると判定した場合には、S 4 に飛ぶ。S 6 は、全タスクを判定した結果、異常状態のタスクは見つからなかったので、タスク終了の SVC を発行して、次に呼ばれるまで何もしない。

【0274】以上、説明した通り、実施例 11 に係る NC 装置の制御ソフトウェア実行システム A 11 にあっては、NC 装置の機能を実現しているタスクに異常が発生したときにそれを検知する機能を備えたものである。

【0275】実施例 11 によれば、NC 装置の制御ソフトウェア実行システム A 11 は、NC 装置の機能を実現

しているタスクに異常が発生したときに、それを検知する機能を備えているため、NC装置に異常が発生したことを直ちに判定することができる。

【0276】〔実施例12〕次に、実施例12について説明する。実施例12に係るNC装置の制御ソフトウェア実行システムA12の全体構成は、実施例11(図9参照)とほぼ同様であるので、その説明を省略する。データ構造についても、実施例11と同じもの(図22、図31参照)を使用するものとする。次に、新たに必要なデータを図33に示す。図33において、360は、

“工具選択使用フラグ”である。このフラグは、NC装置の制御ソフトウェアのうち、新たな加工を行なう処理部でオフして、工具を選択する処理部でオンする。すなわち、このフラグがオンになっていれば、工具選択処理が呼ばれたことを示す。

【0277】また、361は、“工具選択処理スキップフラグ”である。このフラグは、NC装置の制御ソフトウェアのうち、新たな加工を行なう処理部でオフして、以下に述べる、異常判定タスクで工具を選択したときは、異常判定タスクがオンする。本来の工具選択処理は、工具選択処理スキップフラグがオンになっているときは自分では工具の選択を行なわず、異常判定タスクが選択した工具を用いる。362は、本来の工具選択処理と異常判定タスクが選択した工具を伝え合うための“選択工具メモ”である。

【0278】次に、NC装置の制御ソフトウェア実行システムA12の動作について説明する。図34は、NC装置の別工具選択部分の処理手順を示すフローチャートであり、この処理は、実施例11と同様に異常判定タスクにより実行するものとする。T1は、異常判定タスクの先頭で、タスク走行時間テーブル231(図22参照)の、タスク毎の合計チック数の欄の値全ての和を求める処理である。T2は、全てのタスクに対して以下の処理を行なったか否かの判定であり、全てのタスクに対して処理を終了すればT6へ飛ぶ。

【0279】T3は、タスク走行時間テーブル231(図22参照)の、当該タスクの合計チック数の値と、標準データ250(図31参照)の全走行チック数を比較する。当該タスクのタスク走行時間テーブル231(図22参照)の合計チック数の値が、標準データ250(図31参照)の全走行チック数の値に走行時間判定許容度251(図31参照)の値を乗じたものよりも小さいか、等しければ、このタスクは正常であると見なす。

【0280】T4は、T3で、タスクに異常が発生したと判定した場合の処理で、工具選択使用フラグ360(図33参照)がオンになっていなければ、T5に移行する。T5は、T4において、タスクに異常が発生したと判定したときの、通常の処理であり、例えば、オペレータにタスク異常が発生したことを知らせるメッセージ

を送るなど、何らかのエラー処理を実行する。T6は、工具選択使用フラグ360(図33参照)がオンになっている場合の処理であり、次工具を選択する。

【0281】次工具の選択とは、例えば、エンドミルならエンドミル、ボールエンドミルならボールエンドミルというように、同じ種類の工具で工具径が異なる工具を選択することをいう。このとき、例えば、加工モードを設けて、加工速度優先なら工具径のより大きな工具を選択し、精度優先なら工具径のより小さな工具を選択する、などしてもよい。

【0282】このとき、エラーを起こした工具は、選択工具メモ362(図33参照)から知ることができる。T7は、工具が正常に選択できたか否かの判定を行う。T7において、選択する工具がないと判定した場合には、通常のエラー処理部T5へと移行する。T8は、工具が正常に選択できたと判定した場合であって、工具選択使用フラグ360(図33参照)をオフに、工具選択処理スキップフラグ361(図33参照)をオンにして、本来の工具選択処理部に、ここで選択した工具を選択工具メモ362(図33参照)に設定することによって伝達する。その後、加工プログラム解析タスクを起動するなどにより、再び加工を開始する。

【0283】T9は、タスクの走行割合のチェックである。タスク走行時間テーブル231(図22参照)の合計チック数の値を、全タスク分加えて、それで当該タスクの合計チックを割って、当該タスクの走行割合を求める。その値が標準データ250(図31参照)の走行割合判定許容度の下限と上限の間にあれば、このタスクは正常であると見なす。もし、異常があると判定した場合には、T4に移行する。T10は、全タスクを判定した結果、異常状態のタスクは見つからなかったので、タスク終了のSVCを発行して、次に呼ばれるまで何もしない。

【0284】以上、説明したとおり、実施例12に係るNC装置の制御ソフトウェア実行システムA12は、NC装置の自動加工によって、エラーが発生したと判定したとき、工具を変えて再度自動加工を行なうNC装置の制御ソフトウェアを実行するものである。

【0285】実施例12によれば、NC装置の制御ソフトウェア実行システムA12は、NC装置の自動加工によって、エラーが発生したと判定したとき、工具を変えて再度自動加工を実行することにより、エラーの発生によって加工処理が止まる可能性が低くなり、自動的に連続加工ができ、NC装置の連続運転の信頼性が高くなる。

【0286】〔実施例13〕次に、実施例13について説明する。実施例13に係るNC装置の制御ソフトウェア実行システムA13の全体構成は、実施例11(図9参照)の内容とほぼ同様であるので、その説明を省略する。データ構造についても、実施例11と同じもの(図

22, 図31参照)を使用するものとする。

【0287】新たに必要なデータを図35に示す。図35において、370は、“工具速度決定フラグ”である。このフラグは、NC装置の制御ソフトウェアのうち、新たな加工を行なう処理部でオフして、工具の速度を計算する処理部でオンする。すなわち、このフラグがオンになっていれば、工具速度計算処理が呼ばれたことを示す。371は、“工具速度計算処理スキップフラグ”である。このフラグは、NC装置の制御ソフトウェアのうち、新たな加工を行なう処理部でオフして、以下に述べる、異常判定タスクにより工具の速度を変更したときは、異常判定タスクがオンする。

【0288】本来の工具速度計算処理は、工具速度計算処理スキップフラグがオンになっているとき、自分自身では速度の計算は実行せず、異常判定タスクが変更した速度を用いる。また、372は、本来の工具速度計算処理と異常判定タスクが工具速度を伝え合うための“工具速度メモ”である。373は、工具速度変更割合下限値である。この割合を下回っては工具速度は変更できない。374は、工具速度変更割合上限値である。この割合を上回っては工具速度は変更できない。工具速度変更割合下限値373や工具速度変更割合上限値374は予め指定されている。

【0289】次に、NC装置の制御ソフトウェアの実行動作について説明する。図36は、NC装置の別速度計算部分の処理手順を示すフローチャートである。この処理は、実施例11と同様に異常判定タスクで行なうものとする。図において、U1は、異常判定タスクの先頭で、タスク走行時間テーブル231(図22参照)の、タスク毎の合計チック数の欄の値、全ての和を求める処理である。U2は、全てのタスクに対して以下の処理を行なったか否かの判定であり、全てのタスクに対して処理を終了すれば、処理はU6へ移行する。

【0290】U3は、タスク走行時間テーブル231(図22参照)の、当該タスクの合計チック数の値と、標準データ250(図31参照)の全走行チック数を比較する。当該タスクのタスク走行時間テーブル231(図22参照)の合計チック数の値が、標準データ250(図31参照)の全走行チック数の値に走行時間判定許容度251(図31参照)の値を乗じたものよりも小さいか、等しければ、このタスクは正常であると見なす。

【0291】U4は、U3において、タスクに異常が発生したと判定した場合における処理であり、工具速度決定フラグ370(図35参照)がオンになっていなければ、U5に移行する。U5は、U4において、タスクに異常が発生したと判定した場合の通常の処理であり、例えば、オペレータにタスク異常が発生したことを知らせるメッセージを送るなどの、何らかのエラー処理を実行する。

【0292】U6は、工具速度決定フラグ370(図35参照)がオンになっていると判定した場合における処理であり、工具速度を変更する。ここで、速度の変更とは、速度を変えることであるが、このとき、例えば、加工モードを設けて、加工速度優先なら工具速度をある割合ずつ増していく、精度優先なら工具速度をある割合ずつ減らしていくなどしてもよい。このとき、エラーを起こした工具速度は、工具速度メモ372(図35参照)により確認する。

10 【0293】U7は、工具速度が決定できたか否かの判定である。U6において、工具速度が、工具速度変更割合下限値373(図35参照)と工具速度変更割合上限値374(図35参照)との間に入らないときは、通常のエラー処理部U5へと移行する。U8は、工具速度を変更できた場合で、工具速度決定フラグ370(図35参照)をオフに、工具速度計算処理スキップフラグ371(図35参照)をオンにして、本来の工具速度計算処理部に、ここで変更した工具速度を、工具速度メモ372(図35参照)に設定することによって伝える。その後、加工プログラム解析タスクを起動するなどにより、再び加工を開始する。

20 【0294】U9は、タスクの走行割合のチェックである。タスク走行時間テーブル231(図22参照)の合計チック数の値を、全タスク分加えて、それで当該タスクの合計チックを割って、当該タスクの走行割合を求める。その値が標準データ250(図31参照)の走行割合判定許容度の下限と上限との間にあれば、このタスクは正常であると見なす。もし、異常があると判定したときにはU4へ移行する。U10では、全タスクを判定した結果、異常状態のタスクは見つからなかったので、タスク終了のSVCを発行して、次に呼ばれるまで何もない。

30 【0295】以上、説明したとおり、実施例13に係るNC装置の制御ソフトウェア実行システムA13は、NC装置の自動加工によって、エラーが発生したと判定したとき、工具速度を変えて再度自動加工を行なうNC装置の制御ソフトウェアを実行するものである。

【0296】実施例13によれば、NC装置の制御ソフトウェア実行システムA13は、NC装置の自動加工によって、エラーが発生したと判定したとき、工具速度を変えて再度自動加工を行なうことにより、エラーの発生により加工が止まる可能性が低くなり、自動的に連続加工ができNC装置の連続運転の信頼性が高くなる。

40 【0297】[実施例14]次に、実施例14について説明する。実施例14に係るNC装置の制御ソフトウェア実行システムA14の全体構成は、実施例11(図9参照)とほぼ同様であるので、その説明を省略する。

【0298】データ構造についても、実施例11と同じもの(図22、図31参照)を使うものとする。新たに必要なデータを図37に示す。図37において、380

は、"加工プログラム番号メモ"である。この加工プログラム番号メモを見て、加工プログラム解析タスクは、この番号の加工プログラムの解析を実行する。

【0299】また、381は、"加工プログラムスケジューリングテーブル"である。ここへ加工を行なう加工プログラムの"番号"と"回数"などを、予め登録しておく。382は、"加工不能プログラムメモ"である。加工が失敗した加工プログラムの番号を、異常判定タスクが記録しておく。

【0300】次に、NC装置の制御ソフトウェア実行システムA14の動作について説明する。図38は、NC装置の別加工プログラム選択の処理手順を示すフローチャートである。この処理は、実施例11と同様に異常判定タスクにより実行する。V1は、異常判定タスクの先頭で、タスク走行時間テーブル231(図22参照)の、タスク毎の合計チック数の欄の値、全ての和を求める処理である。V2は、全てのタスクに対して以下の処理を行なったか否かの判定であり、全てのタスクに対して処理を終了すれば、処理はV1へ移行する。

【0301】V3は、タスク走行時間テーブル231(図22参照)の、当該タスクの合計チック数の値と、標準データ250(図31参照)の全走行チック数を比較する。当該タスクのタスク走行時間テーブル231(図22参照)の合計チック数の値が、標準データ250(図31参照)の全走行チック数の値に走行時間判定許容度251(図31参照)の値を乗じたものよりも小さいか、等しければ、V9において、このタスクは正常であると見なす。

【0302】V4は、V3において、タスクに異常が発生したと判定した場合における処理であり、異常が発生した加工プログラム番号を、加工不能プログラムメモ382(図37参照)に記録して、加工プログラムスケジューリングテーブル381(図37参照)から次に加工する加工プログラム番号を選択して、加工プログラム番号メモ380(図37参照)に格納する。

【0303】V5は、次に加工する加工プログラムが決定できたか否かの判定である。V4において、加工プログラムスケジューリングテーブル381(図37参照)に登録されている加工プログラムを全て実行し終るとV7の処理に移行する。V6は、次に加工する加工プログラムが決定できた場合であって、加工プログラム解析タスクを起動するなどにより、再び加工を開始する。

【0304】V7は、次に加工する加工プログラムが決定できなかった場合における処理であり、加工不能プログラムメモ382(図37参照)に、データが設定されているか否かにより、加工できなかったプログラムがあるか否かを判定する。V8は、加工できなかったプログラムがあったと判定した場合における処理であり、加工不能プログラムメモ382(図37参照)に基づいて、何らかの手段でオペレータに異常が発生したことを通知

する。また、V10では、全タスクを判定した結果、異常状態のタスクは見つからなかったので、タスク終了のSVCを発行して、次に呼ばれるまで何もしない。

【0305】以上、説明したとおり、実施例14に係るNC装置の制御ソフトウェア実行システムA14は、NC装置の自動加工によって、エラーが発生したと判定したとき、別の加工プログラムにより、再度自動加工を行なうNC装置の制御ソフトウェアを実行するものである。

10 【0306】実施例14によれば、NC装置の制御ソフトウェア実行システムA14は、NC装置の自動加工によって、エラーが発生したと判定したときは、別の加工プログラムで、再度自動加工を行なうことにより、エラーの発生により加工が止まる可能性が低くなり、自動的に連続加工ができるNC装置の連続運転の信頼性が高くなる。

【0307】【実施例15】次に、実施例15について説明する。実施例15に係るNC装置の制御ソフトウェア実行システムA15の全体構成は、実施例2(図9参照)と同じであるので、その説明を省略する。ただし、本実施例にあっては、リアルタイムクロック201(図9参照)の内部レジスタは、202と204の2チャンネルのみを使用する。内部カウンタレジスタ202(図9参照)は、タスクの走行チック数の計測用、内部カウンタレジスタ204(図9参照)は、タスク停止割り込み用として使用する。

【0308】図39は、本実施例において、必要なデータ構造を示したものであり、400は"タスク停止テーブル"である。各タスクは、予め、あるいは、そのタスクが開始させられるときにSVCによって、タスクが起動してから停止するまでの、チック数をこのタスク停止テーブル400の"停止チック数"の欄に登録しておく。また、指定しないタスクはこの値を0としておく。

【0309】210(図10参照)は"タスク開始チックメモ"である。RTOSが、タスクを起動する直前に、リアルタイムクロック201(図9参照)の内部カウンタレジスタ202(図9参照)の値を記録しておく、そして、そのタスクが終了してRTOSに戻って来たときの、内部カウンタレジスタ202(図9参照)との差を求めて、タスク走行時間を求める。411は、"タスク停止割り込みメモ"である。タスク停止割り込みをすぐに処理できないときは、このメモをオンしておいて後に処理する。

【0310】本実施例に係るNC装置の制御ソフトウェア実行システムA15は、タスクを起動するたびに、もし指定されれば、タスク停止テーブル400(図39参照)の停止チック数を、リアルタイムクロック201(図9参照)の内部レジスタ204に設定する。そして、そのタスク停止割り込みが発生すれば、当該タスクを強制的に停止させる。

【0311】もし、タスク停止割り込みが発生する前に、他のタスクによってメインCPU1を占有された場合は、リアルタイムクロック201（図9参照）の内部レジスタは202（図9参照）と、タスク開始チックメモ210（図10参照）から、当該タスクの走行チック数を求め、その分をタスク停止テーブル400（図39参照）の停止チック数から減ずる。このようにして、指定タスクの最大走行チック数を制限することができる。

【0312】次に、NC装置の制御ソフトウェア実行システムA15の動作について、詳細に説明する。最初に割り込みの処理手順について説明する。図40は、タスク停止割り込みの処理手順を示すフローチャートである。W1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。通常この処理はCPUによって自動的に実行される。W2は、割り込まれた処理が、タスク処理中か、それ以外かを判定する処理部分である。W3は、タスク処理中ではないと判定した場合であり、この場合には、タスク停止割り込みメモ411（図39参照）をオンする。W4は、割り込み禁止を解除して、割り込まれた処理へ戻る。

【0313】また、W5は、タスク処理中に割り込みが発生したと判定した場合であって、まず、割り込まれたタスクのTCBを探し出す。W6は、W5において見つけたTCBへ割り込まれたタスクの実行環境を格納する。W7は、タスク停止SVCと同様の当該タスクの終了処理を実行する。例えば、当該TCBのステータスの欄を停止にして、TCB待ち行列から、停止中タスクの待ち行列へつなぎ変えるなどの処理である。W8は、スケジューラへ飛ぶ。

【0314】図41は、タスク停止割り込み以外の、一般的の割り込み処理の手順を示すフローチャートである。X1は、ある割り込みが発生したときにおける割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理はメインCPU1によって自動的に実行される。X2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。

【0315】X3は、ICBに、この割り込み処理（自分自身）の、実行環境を格納する。この中には次に実行する命令のアドレスとしてX8の処理の先頭アドレスをICBに格納することも含まれる。X4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。X5は、X1の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのものである。

【0316】また、X6は、割り込まれた処理が、割り込み処理またはRTOSか、それともタスクの処理中かを判定する。X7は、割り込まれた処理が、割り込み処理またはRTOSを実行中のときであって、このときは

割り込まれた処理に直接戻る。X8は、割り込まれた処理がタスクの処理を実行中の場合であって、このときは、まず割り込まれたタスクのTCBを探し出す。X9は、X8において見つけたTCBへ、割り込まれたタスクの実行環境を格納する。X10は、タスク停止割り込みメモ411（図39参照）がオンか否かを判定する処理である。

【0317】X11は、タスク停止割り込みメモ411（図39参照）がオンのときの処理であり、タスク停止

10 SVCと同様の、当該タスクの終了処理を実行する。例えば、当該TCBのステータスの欄を停止にして、TCB待ち行列から、停止中タスクの待ち行列へつなぎ変える等の処理である。

【0318】X12は、タスク停止割り込みメモ411（図39参照）がオフのときの処理であり、リアルタイムクロック201（図9参照）の内部レジスタは202（図9参照）と、タスク開始チックメモ210（図10参照）との差から、当該タスクの走行チック数を求め、タスク停止テーブル400（図39参照）の停止チック数から減ずる。X13は、X9において作成したTCBを、スケジューラが実行するTCB待ち行列にプライオリティの順に追加する。X14は、スケジューラへ飛ぶ。

【0319】また、X15は、スケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後すぐにこの処理が行なわれるとは限らず、他のICBが処理された後かも知れないが、いずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。X16は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的であり、このリターン命令によって、スケジューラのICB処理部に戻る。

【0320】図42を用いて、本実施例に係るNC装置の制御ソフトウェア実行システムA15のスケジューラの動作を説明する。以下、説明の都合上、本実施例においては、優先順位が最低で、外部に対して何も処理を行なわない、アイドルタスクが常に走っているものとする。

【0321】Y1では、割り込み処理（ICB）の待ち40行列があるか否かを判定する。Y2は、割り込み処理の待ち行列があったときの処理であり、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行なおうとするリストの整合性が保たれなくなるためである。

【0322】Y3は、割り込み処理の待ち行列のリストから先頭のICBを取り除く。Y4は、割り込み禁止を解除する。Y5は、Y3において取り除いたICBから50X3（図41参照）において格納した実行アドレスやレ

ジスタなどの割り込み処理の実行環境を取り出して実行する。これが、X 15 (図41参照)の処理である。Y 6は、X 16 (図41参照)から戻ってくるところであり、再び割り込みを禁止する。Y 7は、Y 3において取り除いたI C BをI C Bプールへ返す。Y 8は、割り込み禁止を解除してからY 1へ戻る。後は割り込み処理待ち行列がある間はY 1からY 7の処理を繰り返す。

【0323】Y 9は、割り込みを禁止する。Y 10は、割り込み処理の待ち行列のリストから、先頭のT C Bを獲得する。Y 11は、当該T C Bのタスクが、タスク停止テーブル400 (図39参照)の停止チック数が、0以上のタスクである場合は、当該タスクのタスク停止テーブル400 (図39参照)の停止チック数の値を、リアルタイムクロック201 (図9参照)の内部レジスタ204に設定する。この結果、設定した数だけのチック数が経過すると、タスク停止割り込みが発生する。Y 12は、割り込み禁止を解除する。Y 13は、Y 10においてタスク処理の待ち行列 (リスト)から獲得した、T C BからX 9 (図41参照)において格納したタスクの実行環境を取り出して実行する。その処理へ飛んでいて、ここへ戻ってこない。

【0324】以上、説明したように、実施例15に係るNC装置の制御ソフトウェア実行システムA15は、通常のプライオリティベースのスケジューリング方式しか持たないR T O Sに、タスク走行時間の制限を行なえる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0325】実施例15によれば、NC装置の制御ソフトウェア実行システムA15は、NC装置機能を実現するタスクの、走行時間による制限を行なえるようになる。そのため、タスクに割り当てた時間を越えて、タスクが走ることがなくなり、システムの構築が容易となる。

【0326】【実施例16】次に、実施例16について説明する。実施例16に係るNC装置の制御ソフトウェア実行システムA16は、加工プログラム解析処理タスクなどを、従来例のように、1ブロック処理する毎にタスクを抜けるのではなく、加工プログラムが存在する限り、解析を続けるように作成しておき、解析の中断は、時間によるものとする。

【0327】実施例16に係るNC装置の制御ソフトウェア実行システムA16の加工プログラム解析タスクの実行手順を図43に基づいて説明する。Z 1は、加工プログラム解析タスクの先頭であり、指定された加工プログラムのブロックを全て解析し終えたか否かを判定する。Z 2は、全ての加工プログラムのブロックの解析を終えたと判定した場合の処理であり、タスク終了S V Cを発行して、再び起動されるまで何もしない。Z 3は、未解析の加工プログラムのブロックが存在する場合の処理であり、ブロックの解析を実行する。以後、全ブロック

クの解析が終了するまで、Z 1の処理を繰り返す。

【0328】以上で述べたように、実施例16に係るNC装置の制御ソフトウェア実行システムA16は、加工プログラム解析タスクを、プログラムがある限り、解析し続けるようにして、解析の中断を時間によって実行するNC装置の制御ソフトウェアを実行するものである。

【0329】実施例16によれば、NC装置の制御ソフトウェア実行システムA16は、加工プログラムの解析を行なうタスクの走行、中断を時間によって行なうようにしたので、加工プログラムのブロックの大きさによる、処理時間の余りが無くなり、特に、微小線分からなる、加工プログラムの解析において処理能力の向上が期待できる。さらにタスクの作成方法として、加工プログラムのブロック毎に、終了するなどの繁雑な処理が不要になる。

【0330】以上のように、この発明の各実施例に係るNC装置の制御ソフトウェア実行システムによれば、NC装置の機能を、分担して実現しているタスクの制御をきめ細かく実行できるので、各タスクの処理時間を最適化することができるため、無駄時間を少なくして、全体の処理時間を早めることができる。また、NC装置の制御ソフトウェア実行システムの異常を、システム 자체が判定できる手段を備えているため、異常状態が発生した時の回避処理も容易となる。

【0331】【実施例17】次に、実施例17について説明する。図44は、実施例17に係る制御ソフトウェア実行システムA17の全体構成を示すブロック図であり、この制御ソフトウェア実行システムA17は、以下の要素より構成されている。1は各部の動作を制御したり、数値制御に必要な演算を実行するメインC P U、2はシステムの各部を結ぶシステムバス、3はNC装置の主要機能を実現する制御ソフトウェアなどを格納するR O M (不揮発性記憶手段)、4はデータの一時記憶やワークエリアなどに用いるR A M (揮発性記憶手段)、5は外部との間でシリアル通信によりデータのやり取りを行なうS I Oインターフェース部である。

【0332】また、図44において、6はNC装置の運転状態を表示したり、NC装置に与えた指令を確認するためにデータ表示を行う表示装置、7はNC装置に対して指令を与えるための入力手段としてのキーボード、8はサーボモータを制御するための指令を演算するサーボ制御部であり、該サーボ制御部8はメインC P U 1とは別の専用C P Uを備えてもよい。

【0333】さらに、図44において、9はサーボ制御部8から受け取った指令を電力増幅してサーボモータや工作機械主軸 (図示せず)を駆動するサーボアンプ、10は工作機械 (図示せず)の加工部を制御するためのサーボモータ、11は工作機械との間でサーボ制御指令以外のデータをやり取りするためのプログラマブルコントローラ部 (以下、P C部という)、12はメインC P U

1に入力される、外部割り込み信号を表している。ここで、外部割り込み信号は電源異常や非常停止などイベント（緊急の出来事）の発生をメインC P U 1に対して通知するための信号である。

【0 3 3 4】また、図4 4において、5 0 0はメインC P U 1に対して入力される、システムクロックである。ここで、システムクロックとは、N C 装置全体を制御するために同期をとるためのクロック信号であり、システム内における各タスクの優先順位を一定間隔毎に調べたり、タスクを指定時間だけ実行を遅らせたりするために、C P U 1に対してある一定周期でタイマ割り込みを入れる。この周期は少なくともマイクロ秒オーダーの十分粒度の小さいものにする。例えば、1 0 Mヘルツのクロック信号を使用する。この場合は、0. 1  $\mu$ 秒(0. 0 0 0 0 0 1秒) 単位でタスクの制御が実行できる。

【0 3 3 5】R T O Sは、システムクロック割り込みがある毎に各タスクの状態を調べ、実行中のタスクを止めて、別のタスクを実行させたり（これをスケジューリング、あるいは、ディスパッチという）する。

【0 3 3 6】図4 4において、5 0 1はR A M 4内に格納されたシステムクロック数メモである。このシステムクロック数メモ5 0 1は所定の条件のときに、システムクロック5 0 0の入力がある毎にR T O Sが減らしていく。5 0 2はR A M 4内に格納されたシステムクロック数メモ有効フラグである。このメモが有効を示す値に設定されている場合のみ前記システムクロック数メモ5 0 1を有効と見なすものである。

【0 3 3 7】また、5 0 8はR A M 4内に格納されたディスパッチクロック数である。ここに設定されたシステムクロックの回数分だけシステムクロックの入力があったときに、ディスパッチなど従来の実施例で行っていたのと同様のR T O Sの処理を実行する。このディスパッチクロック数5 0 8の設定は、通常電源投入時に1回だけ行えばよいが、システムが稼働している最中に変更することにより、スケジューラの呼び出し頻度を変えることも可能である。

【0 3 3 8】さらに、5 0 9はR A M 4内に格納されたディスパッチクロック数メモであり、システムクロック5 0 0の入力がある毎に、この値を減らしていく、0になったときにディスパッチなどの従来の実施例で行ったのと同様のR T O Sの処理を実行する。

【0 3 3 9】次に、上記の構成を有するN C 装置の制御ソフトウェア実行システムA 1 7の動作について説明する。

【0 3 4 0】各タスクは、予め、あるいは、そのタスクが開始させられるときにS V C（スーパーバイザーコール、または、サービスコール）によってその実行周期を登録する。ここで、S V Cとは、タスクがR T O Sに対して何らかのサービスを要求するときの総称である。何らかのサービスとは、例えば、何らかの条件が成立する

まで自分自身の実行を停止させたり、他のタスクと通信を行なったり、他のタスクを起動させたり（実行を開始させたり）、停止させたりすることである。

【0 3 4 1】このデータは、R T O Sが、例えば、上記図2に示すような“タスク走行周期テーブル”5 0に登録する。図2は、タスク毎に“走行周期”と“繰り返し回数”を有するデータ構造である。ここで、走行周期とは、タスクの次に走るまでのクロック数の値であり、また、繰り返し回数とは、何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば $\infty$ を示すものとし、その場合にあっては、タスクは繰り返し起動するものとする。このタスク走行周期テーブル5 0を用いて、R T O Sは、タスクに起動がかかるたびに図3に示すような“タスク走行待ちリスト”5 1を作成する。

【0 3 4 2】図3において、タスク走行待ちリスト5 1の中の各走行待ちブロック5 2は、それぞれ、次の走行待ちブロックの位置を示す“リンク”，“タスク名”，今現在からどれだけの時間が経過した後に走らなくてはならないかの“走行予定クロック数”，同時に走らせなくてはならないタスク名を示す“同時タスク名”，それにT C Bの位置を示す“T C B ポインタ”等の要素から構成されるブロックである。

【0 3 4 3】以下に、図4 5のフローチャートを用いて具体的な動作について説明する。説明の都合上、システムに入力するシステムクロック5 0 0の周波数は1 0 Mヘルツ、すなわち、0. 1  $\mu$ 秒毎にシステムクロック5 0 0が入力されるものとする。また、説明中、リストを操作している最中など必要な処理中は、割り込みを禁止しているものとする。さらに、走行待ちブロック5 2を作成する場合には、タスク名としては当該タスクを示す名前を入れておくものとする。

【0 3 4 4】図4 5は、R T O Sが、タスク走行待ちリスト5 1を作成するときの手順を示したものである。図において、まず、B B 1では、要求のあった走行周期を内部クロック数に変換する。例えば、要求が1 0 m秒であれば、1 0 0, 0 0 0 (1 0 0, 0 0 0  $\times$  0. 1  $\mu$ 秒 = 1 0 m秒) と変換する。これをR T (request time) とする。次に、B B 2では、現在の走行待ちリストに走行待ちブロックがあるか否かを判定し、走行待ちブロックがないと判定した場合には、B B 3に処理が移行し、反対に、走行待ちブロックがあると判定した場合には、B B 6に処理が移行する。

【0 3 4 5】B B 3は、走行待ちリスト5 1に走行待ちブロック5 2がなかった場合における処理であり、走行待ちブロック5 2を作成し、B B 4では、走行待ちリスト5 1の先頭に走行待ちブロック5 2を置く。その後、B B 5では、走行待ちリスト5 1の先頭ブロックの走行予定クロック数を、システムクロック数メモ5 0 1へ格納する。B B 2 1では、システムクロック数メモ有効フ

ラグ 502 へ有効を示す値を設定する。

【0346】次に、BB6 は、走行待ちリスト 51 に走行待ちブロック 52 があると判断した場合における処理であり、最終タスク走行時間 LT を 0 にして、タスクインデックス TI を 1 にする。ここで、LT とは当該タスクを走らせるのは現在からどれだけのクロック数かを計算するための変数であり、TI は走行待ちリスト中で当該タスクを示すインデックスである。

【0347】その後、BB7 では、現在のシステムクロック数メモ 501 の値を読み込んで、先頭の走行待ちブロック 52 の走行予定クロック数を入れる。次に、BB8 において、走行待ちリスト 51 内の、全ての走行待ちブロック 52 を調べ終ったか（たどったか）否かを判定する。

【0348】その結果、BB8 において、全ての走行待ちブロック 52 を調べ終っていないと判断した場合には、BB9 に処理が移行する。BB9 では、走行待ちブロック 51 が、走行待ちリスト 51 中にあったと判断した場合の処理であり、走行待ちリスト 51 の中から、TI の示す走行待ちブロック 52 の、走行予定クロック数を取り出して LT に加える。

【0349】その後、BB10 において、RT と LT を比較する。BB11 は、RT と LT が等しい場合 (RT = LT) の処理であり、走行待ちブロック 52 を作成する。このとき、走行予定クロック数は 0 にしておく。BB12 は、BB11 において作成した走行待ちブロック 52 を、走行待ちリスト 51 中の TI の示すブロックの同時タスク名へつなぐ。ここで、つなぐとは、例えば、BB11 において作成したブロックのアドレスを TI が示す走行待ちブロック 52 の同時タスク名の場所に置くなど、要するに BB11 において作成したブロックを、直ちに見つけられるような配置に設定することである。

【0350】BB13 は、RT が LT より大きな場合 (RT > LT) の処理であり、TI の値を、1 プラスして、BB8 へ戻り同じ処理を繰り返す。

【0351】BB14 は、RT が LT より小さな場合 (RT < LT) の処理であり、走行待ちブロック 52 を作成する。このとき、走行予定クロック数は、RT - LT + (TI が示すより 1 つ前の走行待ちブロック 52 の走行予定クロック数) の値にする。

【0352】その後、BB15 では、TI が示すブロックの走行予定クロック数を LT - RT とし、BB16 では、TI が示す走行待ちブロック 52 とその前のブロックとの間に BB14 において作成した走行待ちブロック 52 を挿入する。

【0353】その後、BB17 では、BB16 で挿入したブロックが、走行待ちリスト 51 の先頭か否かを判定し、BB18 において、先頭であると判定した場合には、走行待ちリスト 51 の先頭ブロックの走行予定クロ

ック数を、システムクロック数メモ 501 に格納する。

【0354】BB19 では、BB8 において、全ての走行待ちブロックを調べ終ったと判定した場合に、走行予定クロック数が RT - LT の走行待ちブロック 52 を作成し、その後、BB20 では、BB19 において作成した走行待ちブロック 52 を走行待ちリスト 51 の最後に挿入する。

【0355】上記実施例における図 5 は、走行待ちリスト 51 の例である。この例では現在から 30 クロック後にタスク A を起動して、さらにそれから 20 クロック後（すなわち、現在から 50 クロック後）にタスク B を起動することを示している。このとき、タスク C を 35 クロック後に起動する要求が RTOS に対して発せられたとする、図 4 5 に示したフローチャートの手順にしたがって走行待ちリストは図 6 に示すようになる。

【0356】図 4 5 に示した BB5 や、BB18 において走行待ちリスト 51 の先頭ブロックの走行予定クロック数を、システムクロック数メモ 501 に格納し、その指定したクロック数が経過した後、後述するシステムクロック割り込みからタスク起動処理が呼ばれる。このタスク起動処理について図 4 6 を用いて説明する。

【0357】図 4 6 において、CC1 では、走行待ちリスト 51 の先頭の走行待ちブロック 52 を走行待ちリスト 51 から切り離す処理を実行し、その後、CC2 では、走行待ちリスト 51 が空になったか否かを判定する。また、CC3 は、走行待ちリスト 51 が空になったと判定した場合の処理であり、システムクロック数メモ有効フラグ 502 を無効にすることにより、以降タスク起動処理が呼ばれないようにして、処理を修了する。

【0358】また、CC4 は、CC2 において、走行待ちリスト 51 に走行待ちブロック 52 が残っていると判定した場合の処理であり、先頭の走行待ちブロック 52 の走行予定クロック数を、システムクロック数メモ 501 に設定する。その後、CC5 では、CC1 において切り離した走行待ちブロック 52 から起動するタスクを取り出して、そのタスクの TCB を TCB 待ち行列の先頭に配置する。

【0359】その後、CC6 では、今走らせたタスクの、タスク走行周期テーブル 50 (図 2 参照) の周期回数を判定する。CC7 は、走行回数が  $\infty$  (負) の場合の処理であり、このタスクの走行待ちブロック 52 の、走行予定クロック数を、タスク走行周期テーブル 50 の走行周期に変え、走行待ちブロックを走行待ちリストに追加する。

【0360】CC8 は、走行回数が正の整数だった場合の処理であり、走行予定回数を 1 減らす。その後、CC9 では、当該タスクに対して図 4 に示した処理により、このタスクの走行待ちブロック 52 を作成して走行待ちリストに追加する。CC10 は、走行回数が 0 の場合の処理であり、この場合、何の処理も実行せずに処理を終

了する。

【0361】次に、システムクロック割り込み処理について、図47を用いて説明する。DD1は、システムクロック割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する。ただし、この処理は、通常メインCPUにより割り込みが発生したときに自動的に実行される。

【0362】次に、DD20はディスパッチクロック数メモ509から1減じる処理である。DD14はシステムクロック数メモ有効フラグ502が有効になっているか否かを判定する判定処理であり、有効になっていないと判定した場合には、後述するDD21の処理へ移行し、反対に、有効になっていると判定した場合には、DD17の処理を実行する。

【0363】DD21はディスパッチクロック数メモ509が0になったか否かを判定する処理であり、DD22はディスパッチクロック数メモ509が0になった場合の処理で、改めてディスパッチクロック数508の値で初期化する。その後、DD2からのディスパッチ処理を実行する。

【0364】DD15はディスパッチクロック数メモ509が0になっていない場合の処理で、割り込み禁止を解除する処理である。DD16は割り込まれた処理にスケジューラを介さずに直接戻る処理である。DD17は割り込みクロック数メモ501から1減ずる処理である。

【0365】DD18は割り込みクロック数メモ501が0に等しくなったか否かを判定するための処理であり、この結果が0に等しくない場合には、まだタスク起動処理を呼ばないため、ディスパッチを行うか否かを判定するためにDD21へ移行する。DD19は、図46において説明したタスク起動処理を呼ぶための処理である。タスク起動処理から戻ってるとDD23の処理へと移行する。DD23の処理はディスパッチクロック数メモ509が0になったか否かを判定する処理であり、DD24はディスパッチクロック数メモ509が0になった場合の処理で、改めてディスパッチクロック数508の値で初期化する。

【0366】また、DD2は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する。

【0367】その後、DD3では、ICBに、自分自身の割り込み処理の、実行環境を格納保存する。この中には、次に実行する命令のアドレスとしてDD12の処理における先頭アドレスをICBに格納することも含まれる。また、DD4では、今作成したICBを、スケジューラが実行するICB待ち行列に追加する。その後、DD5は、DD1の割り込みの禁止を解除する。これ以降は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのもので

ある。

【0368】次に、DD6は、割り込まれた処理が、割り込み処理あるいはRTOSか、それともタスクの処理中かを判定する。その結果、DD7は、割り込まれた処理が割り込み処理またはRTOSを実行中であると判断した場合であって、このときは割り込まれた処理に直接戻る。反対に、DD8は、割り込まれた処理が、タスクの処理を実行中であると判断した場合であって、このときは、まず割り込まれたタスクのTCBを探し出す。その後、DD9では、DD8において見つけたTCBへ割り込まれたタスクの実行環境を格納保存する。

【0369】さらに、DD10では、作成したTCBを、スケジューラが実行するTCB待ち行列に優先順位の順番に追加する。また、DD11では、スケジューラへ飛び、スケジューラからDD12に処理が戻ってくるが、スケジューラへ飛んだ後すぐにこの処理が実行されるとは限らず、他のICBが処理された後に戻る場合もある。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理が異なる。さらに、DD13は、割り込み処理の終了であり、通常のサブルーチンへのリターン命令が一般的であり、このリターン命令によって、スケジューラICB処理に戻る。

【0370】以上の説明の中では、また以後の割り込みの説明において、従来例のように、割り込み自身のアドレスをリストに登録しておいて、多重割り込み（割り込み処理中の他の割り込み）を許可する処理は、従来例と同様であるので、その説明は省略する。

【0371】上記のとおり、実施例17に係るNC装置の制御ソフトウェア実行システムA17は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSのシステムクロックを細かい時間粒度による制御を可能にし、なおかつ、毎回のシステムクロック割り込みによるスケジューラ呼び出しを必要なときだけに制限して、RTOSにオーバーヘッドもなく、タスク起動の制御が実行される機能を追加し、NC装置の制御ソフトウェアを実行するものである。

【0372】実施例17によれば、NC装置の制御ソフトウェア実行システムA17は、NC装置機能を実現するタスクの時間による制御を小さな時間単位で行なえるようになり、さらにタスクを起動する周期を簡単に指定できる。そのため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間で制御できるので、システムの構築が容易になるとと共にNC装置の処理速度が向上する。

【0373】【実施例18】次に、実施例18の構成について説明する。実施例18に係るNC装置の制御ソフトウェア実行システムA18の全体構成を示す図18の内容は、実施例17において示した図44とほぼ同様である。ただし、実施例17にあっては、システムクロック数メモ501しかなかったが、本実施例では他に、積

算システムクロック数メモ 503 と走行時間警告用システムクロック数メモ 505 を備えていることが異なる。

【0374】以下、実施例 18 に係る NC 装置の制御ソフトウェア実行システム A18 の構成を詳細に説明する。なお、この制御ソフトウェア実行システム A18 の 1~12, 500, 508, 509 は、上記実施例 17 の図 44 に示したものと同様であるので、その説明を省略する。

【0375】図 48において、501 は RAM 4 内に格納された基準時間割り込み用のシステムクロック数メモである。このシステムクロック数メモ 501 はシステムクロック 500 の入力がある毎に R T O S が減らしていく。502 は RAM 4 内に格納されたシステムクロック数メモ有効フラグであり、システムクロック数メモ 501 が有効か否かを示す。503 は RAM 4 内に格納されたタスク走行時間計測用の積算システムクロック数メモであり、システムクロック 500 の入力がある毎に R T O S が増加させる。505 は RAM 4 内に格納された走行時間警告用に用いる、走行時間警告用システムクロック数メモである。

【0376】次に、NC 装置の制御ソフトウェア実行システム A18 の動作について説明する。図 49 は、本実施例において必要なデータ構造を示したもので、510 は“タスク開始システムクロックメモ”であり、R T O S が、タスクを起動する直前に図 48 に示した積算システムクロック数メモ 503 の値を記録しておき、その後、タスクが終了して R T O S に戻ってきたときの積算システムクロック数メモ 503 との差を求ることによって、タスクの走行時間を求めるためのものである。

【0377】また、511 は“基準周期”を格納しておく領域である。システムに対して予め 1 つの基準周期を登録しておく。この周期毎に各タスクはどれくらい走行するかを指定する。この指定は時間でもクロック数でも構わないが、ここでは説明の都合上、クロック数に変換しておくものとする。

【0378】さらに、512 は“タスク走行状態メモ”である。各タスクは予め、あるいは、そのタスクが開始させられるときに S V C によって、その実行の割合を登録する。例えば、タスク毎の指定走行周期、すなわち、タスク A が 10 %、タスク B が 5 %、タスク C が 20 %、基準周期が 20 m 秒のとき、20 m 秒毎にタスク A が 2 m 秒、タスク B が 1 m 秒、タスク C が 4 m 秒走ればよい。

【0379】ここで、実施例 17 と同様、図 48 におけるシステムクロック 500 は、10 M ヘルツとする。すなわち、基準となる最小単位時間は 0.1 μ 秒とする。A, B, C 各タスクはクロック数 200000 クロックの間に 20000, 1000, 4000 クロック分走ればよいことになる。このクロック数をタスク走行状態メモ 512 の“走行クロック数”の欄に格納する。

ここで、走行する割合を指定しないタスクに対してはこの欄は 0 にしておく。

【0380】また、“走行残りクロック数”の欄は、1 回の基準周期が終了する毎に、走行クロック数の欄をコピーし、タスクが走ったクロック数だけ減算していく。1 回の基準周期が終る毎に、すべてのタスクの、この欄の内容が 0 になっていればよい。当然走行クロック数の合計は基準周期よりも少なくてはいけない。そのエラーチェックは、例えば、タスク状態メモに新たに登録する毎に行なえばよい。ただし、以下の説明において、このエラーチェックについては省略する。

【0381】次に、“繰り返し回数”の欄に設定するのは、何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば∞を示すとする。その場合には、当該タスクは繰り返し起動するものとする。

【0382】また、“走行禁止フラグ”の欄は、スケジューラが用いる。このフラグがオンになっているタスクはスケジューリングの対象から外す。すなわち、このフラグがオンのタスクはプライオリティが高くても走ることができない。

【0383】次に、基準周期リセット処理について説明する。制御ソフトウェア実行システム A18 は、稼働始めたときなど走行割合を保証するタスクが動作し始める前に、予めシステムクロック数メモ 501 に、基準周期 511 の値を設定しておく。その指定したクロック数が経過した後、後述するシステムクロック割り込みから基準周期処理が呼ばれる。

【0384】図 50 は、基準周期リセット処理の手順について説明したフローチャートである。基準周期処理は、基準周期毎のシステムおよび全タスクの再初期化処理と考えられる。E E 2 では、タスク走行状態メモ 512 の走行残りクロック数の合計を求める。E E 3 では、その合計が 0 以上か否かを判定し、0 以上ではないと判定した場合には、E E 4 において何らかのエラー処理を行なうが、通常は発生しないのでここでは、その説明を省略する。

【0385】反対に、E E 3において、合計が 0 以上であると判定した場合には、E E 5においてタスク走行状態メモ 512 の走行クロック数の欄の値を走行残りクロック数の欄に、タスク毎にコピーする。その後、E E 6 では、再びシステムクロック数メモ 501 に対して基準周期 511 の値を設定する。

【0386】その後、E E 7 では、タスク走行状態メモ 512 のすべてのタスクの走行禁止フラグを、オフにする処理を行う。この処理により、再びすべてのタスクがスケジューリング対象になる。さらに、E E 8 では、基準周期 511 からタスク走行状態メモ 512 の走行残りクロック数の合計を減算した値を走行時間警告用システムクロック数メモ 505 に設定する。その結果、毎回の

基準周期に残り時間が周期を保証したタスクだけを走らせる時間しかなくなつたときに走行時間警告処理が呼ばれる。この後、呼び出された処理に戻る。

【0387】図51は、走行時間警告処理の手順について説明したフローチャートである。FF2は、タスク走行状態メモ512の走行残りクロック数が、0または負の全てのタスクの走行禁止フラグをオンにする。

【0388】FF3では、全ての走行禁止フラグがオフであるTCBが、全ての行禁止フラグがオンのTCBの前に来るよう、タスク処理の待ち行列を作り直す。すなわち、走行禁止フラグがオンのタスクの方がフラグがオフのタスクよりも前に走るようにする。

【0389】図52は、システムクロック割り込み処理の手順を示すフローチャートである。以下の説明において、有効フラグはシステムクロック数メモ有効フラグ502に関するものしか説明していないが、走行時間警告用システムクロック数メモ505に関しても同様の有効フラグが必要である。本フローチャートの中では、走行時間警告用システムクロック数メモ505用の有効フラグに対する処理は、システムクロック数メモ有効フラグ502と同様であるので、その説明は省略する。

【0390】GG1は、ある割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理はメインCPUによって自動的に実行される。GG20は積算システムクロック数メモ503に1を加える処理である。GG21はskipフラグをオフにする処理である。ここで、skipフラグとは、RAM4上に設けた今回のシステムクロック割り込みでスケジューリングなどの処理を実行するか否かを示すために用いるフラグである。

【0391】GG22はシステムクロック数メモ501から1減ずる処理である。GG23はGG22の処理の結果システムクロック数メモ501が0になったか否かを判定する処理である。GG24はシステムクロック数メモ501が0になった場合の処理で、図50に示した基準周期リセット処理を呼ぶ。GG25はskipフラグをオフにする処理である。GG26は走行時間警告用システムクロック数メモ505から1減ずる処理である。

【0392】また、GG27はGG26の処理の結果、走行時間警告用システムクロック数メモ505が0になったか否かを判定する処理である。GG28は走行時間警告用システムクロック数メモ505が0になった場合の処理であつて図51に示した走行時間警告処理を呼ぶ。GG29はskipフラグをオフにする処理である。GG32はディスパッチクロック数メモ509から1減算する処理である。GG33はGG32の結果、0か否かを判定する処理である。

【0393】さらに、GG34はディスパッチクロック数メモ509をディスパッチクロック数508で再初期

化する処理である。GG35はskipフラグをオフにする処理である。GG30はskipフラグがオンかオフかを判定する処理であり、オフの場合には、スケジューリング処理等を実行するため、処理はGG2に移行する。GG31はskipフラグがオンの場合の処理であり、割り込まれた処理に直接復帰するものである。

【0394】GG2は、未使用のICBをICBプール(未使用ICBを保持しておく場所)から1つ獲得する。GG3は、ICBに、この割り込み処理(自分自身の)の、実行環境を格納(保存)する。この中には、次に実行する命令のアドレスとして後述するGG15の処理の先頭アドレスをICBに格納することも含まれる。

【0395】GG4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。GG5は、GG1の割り込みの禁止を解除する。これ以後は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためのものである。

【0396】GG6は、割り込まれた処理が、割り込み処理か、あるいは、RTOSか、それともタスクの処理中かを判定する。GG7は、割り込まれた処理が割り込み処理またはRTOSを実行中の場合で、このときは割り込まれた処理に直接戻る。GG8は、割り込まれた処理がタスクの処理を実行中のときで、このときには、まず、割り込まれたタスクのTCBを探し出す。GG9は、GG8において見つけたTCBへ割り込まれたタスクの実行環境を格納する。

【0397】GG10は、現在処理を行なっているタスクが、走行する割合を保証する対象のタスクであるか否かを判定する処理である。具体的には、タスク走行状態メモ512の走行残りクロック数の欄が1以上のタスクか否かを判定する。GG11は、GG10において対象と判定した場合における処理であり、当該タスクを起動したときの積算システムクロック数メモ503の値から、記録してあるタスク開始システムクロックメモ510の値を減じた当該タスクの走行クロック数を、タスク走行状態メモ512の当該タスクの走行残りクロック数の欄から減算する。

【0398】GG12は、GG11において減算した値(当該タスクの走行クロック数)を、走行時間警告用システムクロック数メモ505の値に加える。走行周期を保証したタスクが、走行し終った分だけ、走行周期を保証したタスク以外のタスクを走らせるための時間が増えるわけである。

【0399】GG13は、作成したTCBを、スケジューラが実行するTCB待ち行列に追加する。このとき、タスク走行状態メモ512の走行禁止フラグの欄が1つのタスクでもオンになつていれば、TCB待ち行列において、走行禁止フラグがオフのタスクのTCBは、走行禁止フラグがオンのタスクのTCBよりも前につなぐ。

【0400】GG14は、スケジューラへ飛ぶ。GG15は、スケジューラから戻ってきたときの処理であり、スケジューラへ飛んだ後すぐにこの処理が実行されるとは限らず、他のICBが処理された後の場合もあるが、いずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理内容が異なる。GG16は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的であり、このリターン命令によって、スケジューラのICB処理部に戻る。

【0401】次に、RTOSのスケジューラの動作を図53を用いて説明する。以下、説明の都合上、本実施例においては、優先順位が最低であって、外部に対して何も処理を行なわない、アイドルタスクが常に走っているものとして説明を行なう。

【0402】HH1は、割り込み処理（ICB）の待ち行列があるか否かを調べる処理である。HH2は、HH1において、割り込み処理の待ち行列があったと判断した場合の処理であり、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加をしようとリストの整合性が保たれなくなるためである。

【0403】HH3は、割り込み処理の待ち行列のリストから先頭のICBを取り除く。HH4は、割り込み禁止を解除する。HH5は、HH3において取り除いたICBからGG3において格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して、それを実行する。これが、GG15の処理である。HH6は、GG16から戻ってくるところであり、再び割り込みを禁止する。HH7は、HH3において取り除いたICBをICBプールへ返す。HH8は、割り込み禁止を解除してからHH1へ戻る。後は割り込み処理待ち行列がある間はHH1からHH7の処理を繰り返す。

【0404】HH9は、タスク処理の待ち行列があったときの処理で、その待ち行列を実行するために最初に割り込みを禁止する。HH10は割り込み処理待ちの行列のリストから先頭のTCBを取り除く。HH11は当該TCBのタスクが走行割合を保証するタスクである場合には、タスクを開始し、システムクロック数メモ501に積算システムクロック数メモ503の値を保存しておく。

【0405】HH12は割り込み禁止を解除する。HH13は、HH11でタスク処理の待ち行列（リスト）から取り除いたTCBからGG9で格納したタスクの実行環境を取り出して実行する。そのときは、その処理へ飛んでいってここへは再び戻ってこない。

【0406】以上で述べたように、実施例18に係るNC装置の制御ソフトウェア実行システムA18は、通常のプライオリティベースのスケジューリング方式しか持

たないRTOSに、タスク走行時間の保証を与える制御を実行できる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0407】上記実施例18によれば、NC装置の制御ソフトウェア実行システムA18は、NC装置機能を実現するタスクの走行時間による制御を実行することができる。そのため、タスクに割り当てた時間の無駄を少なくできる。さらに、タスクをその走行時間において制御できるので、システムの構築が容易になる。

【0408】【実施例19】次に、実施例19の構成について説明する。実施例3に係るNC装置の制御ソフトウェア実行システムA19の全体構成を示す図54の内容にあっては、実施例18において示した図48の内容とほぼ同様である。以下、構成について詳細に説明する。この制御ソフトウェア実行システムA19の1~12、500、508、509は、上記実施例17の図44に示した構成と同様であるので、その説明を省略する。

【0409】この制御ソフトウェア実行システムA19は、次の構成要素がさらに加えられている。すなわち、タスクの走行を修了させるために用いるタスク連続走行終了クロック数メモ506と、上記タスク連続走行終了クロック数メモ有効フラグ507である。

【0410】次に、実施例19の動作について説明する。図55、図56は、タスクの連続走行を保証するのに必要なデータ構造を示したもので、図55は、“タスク終了クロック数”のリスト構造である。520は、“タスク終了クロックブロック”を表している。タスク終了ブロックの構成要素は、タスク終了ブロックを連結するための“リンク（ポインタ）”，“タスク名”，“終了クロック数”などである。今までの説明と同様本実施例の説明でも内部的には全てクロック数により時間の制御を行なうものとする。

【0411】図56は、“タスク連続走行保証テーブル”521の内容を示している。このテーブルには“タスク名”と“連続走行保証クロック数”と“実行中フラグ”的欄があり、連続走行保証クロック数の値はシステム起動時、またはSVCにより設定する。連続走行を指定しないタスクに対してはこの欄は0にしておく。実行中フラグ欄は現在連続走行を保証されたタスクが走っているときはオン、それ以外のときはオフとする。このフラグは同時に2つ以上オンすることは無い。

【0412】このテーブルはタスク起動のSVCにおいて、そのパラメータとして指定することにすれば必要はなくなるが、以下の説明の都合上、ここではこのテーブルを用いることにする。

【0413】RTOSのスケジューラは、連続走行が保証されているタスクが走行しているときに、割り込みが発生しても割り込み処理の終了後の再スケジューリング

は行わないことによって、タスクの連続走行を保証する。連続走行が保証されているタスクの実行を終了させるのは、所定の条件が成立したとき、例えば、“連続走行終了処理”を呼んだときである。

【0414】R T O S はタスク起動の S V C が発行されたときは、まずそのタスクが連続走行を保証されたタスクか否かを判定して、もし該当するタスクなら、タスク連続走行保証テーブル 521 の連続走行クロック数をタスク連続走行終了クロック数メモ 506 に代入する。システムクロックが入力される毎にこの値から 1 ずつ減じていって、0 になったときにタスク連続走行終了処理を呼ぶ。

【0415】最初に、タスク連続走行終了処理について説明する。図 57 は、タスク連続走行終了処理について示したフローチャートである。ここで、I I 2 では現在走っている連続走行を保証されたタスクの T C B を探し出して、現在のタスクの実行環境を T C B へ保存する。I I 3 ではタスク連続走行保証テーブル 521 内の、当該タスクの実行中フラグをオフする処理である。I I 4 では同様のタスク処理の待ち行列を優先順位に応じて作り直す。

【0416】次に、システムクロック割り込み処理の手順について図 58 のフローチャートを用いて説明する。まず、J J 1 では、割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理は C P U によって自動的に行われる。J J 2 0 は、タスク連続走行終了クロック数メモ有効フラグ 507 が有効か否かの判定処理である。

【0417】J J 2 1 は、タスク連続走行終了クロック数メモ有効フラグ 507 が有効でなかった場合の処理で、ディスパッチクロック数メモ 509 から 1 を減算する。J J 2 2 は、上記ディスパッチクロック数メモ 509 が 0 になったか否かを判定する処理である。J J 2 3 は、ディスパッチクロック数メモ 509 の値をディスパッチクロック数 508 で再処理化する処理である。J J 2 4 は、割り込み禁止の解除処理である。J J 2 5 は、割り込まれた処理に直接戻る処理である。

【0418】また、J J 2 6 は、タスク連続走行終了クロック数メモ 506 から 1 減算する処理である。J J 2 7 は、J J 2 6 の結果を判定する処理であり、0 でなければ J J 2 1 へ飛ぶ。J J 2 8 は、図 57 において説明したタスク連続走行終了処理を呼ぶ処理である。

【0419】J J 2 9 は、ディスパッチクロック数メモ 509 から 1 を減算する。J J 3 0 は、上記ディスパッチクロック数メモ 509 が 0 になったか否かを判定する処理である。この結果が、0 でない場合は、J J 2 へ飛ぶ。J J 3 1 は、ディスパッチクロック数メモ 509 の値をディスパッチクロック数 508 で再初期化する処理である。この後 J J 2 へ飛ぶ。

【0420】J J 2 は、未使用の I C B を I C B プール

(未使用 I C B を保持しておく場所) から 1 つ獲得する。J J 3 は、I C B に、この割り込み処理（自分自身の）の、実行環境を格納する。この中には次に実行する命令のアドレスとして J J 1 3 の処理の先頭アドレスを I C B に格納することも含まれる。

【0421】J J 4 は、今作成した I C B を、スケジューラが実行する I C B 待ち行列に追加する。J J 5 は、J J 1 の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためである。

【0422】J J 6 は、割り込まれた処理が割り込み処理または R T O S か、それともタスクの処理中かを判定する。J J 7 は、割り込まれた処理が割り込み処理または R T O S を実行中のとき、このときは割り込まれた処理に直接戻る。J J 8 は、割り込まれた処理がタスクの処理を実行中のとき、このときはまず、割り込まれたタスクの T C B を探し出す。

【0423】J J 9 は、今処理を行っているタスクが、連続走行を保証する対象のタスクであるか否かを判定する処理である。具体的には、タスク連続走行保証テーブル 521 (図 56 参照) の連続走行保証クロック数の欄が 1 以上のタスクか否かを判定する。連続走行保証の対象タスクの場合は、以下の J J 1 0, J J 1 1 の処理は行わないで、J J 1 2 へ飛ぶ。J J 1 0 は、連続走行保証の対象外のタスクの場合の処理で、J J 8 で見つけた T C B へ割り込まれたタスクの実行環境を格納する。

【0424】J J 1 1 は、T C B 待ち行列を優先順位に応じて作り直す。J J 1 2 は、スケジューラへ飛ぶ。J J 1 3 は、スケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後すぐにこの処理が行われるとは限らず、他の I C B が処理された後かも知れないがいずれはここへ戻ってくる。

【0425】ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。J J 1 4 は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が多い、このリターン命令によって、スケジューラの I C B 処理部に戻る。

【0426】以上で述べたように、実施例 19 に係る N C 装置の制御ソフトウェア実行システム A 19 は、通常のプライオリティーベースのスケジューリング方式しか持たない R T O S に、タスクの連続した走行時間の保証を与える制御を実行できる機能を追加した方式であり、この方式に基づいて N C 装置の制御ソフトウェアを実行するものである。

【0427】上記実施例 19 によれば、N C 装置の制御ソフトウェア実行システム A 19 は、N C 装置機能を実現するタスクに連続した走行時間を保証できるようになる。そのため、タスクに割り当てた時間の無駄を少なくできる。さらに、タスクをその走行時間において制御できるので、システムの構築が容易に行える。

【0428】〔実施例20〕次に、実施例20の構成について説明する。実施例20に係るNC装置の制御ソフトウェア実行システムA20の全体構成を示す図59の内容は、実施例17において示した図44の内容とほぼ同様である。

【0429】図59に示したRAM4において、501はシステムクロック数メモ、503は積算システムクロック数メモ、505はタスクの走行時間警告用の走行時間警告用システムクロック数メモ、506はタスク連続走行終了クロック数メモである。なお、508は、図44と同様にシステムクロック何回に1回ディスパッチ処理を呼ぶかを示すディスパッチクロック数であり、509はディスパッチクロック数メモである。

【0430】次に、NC装置の制御ネットワーク実行システムA20の動作について説明する。実施例17と同様に、各タスクは、予め、あるいはそのタスクが開始させられるときにSVCによってその実行周期を登録する。

【0431】このデータは、RTOSが、例えば、図2に示したような“タスク走行周期テーブル”50に登録する。図2は、タスク毎に“走行周期”と“繰り返し回数”を持ったデータ構造である。走行周期はタスクの次に走るまでのクロック数の値であり、繰り返し回数は何回そのタスクを起動するかを示す回数である。例えば、この値がマイナスの値ならば∞を示すとする。その場合はずっとそのタスクは繰り返し起動するものとする。他のタスク走行周期テーブル50を用いて、RTOSは、タスクに起動かかかるたびに図3のような“タスク走行待ちリスト”51を作成する。

【0432】図3において、リストの中における走行待ちブロック52は、それぞれ“タスク名”と、今現在からどれだけの時間の経過後に走らなくてはならないかの“走行予定クロック数”，それに同時に走らせなくてはならないタスク名を示す“同時タスク名”から成るブロックである。

【0433】このタスク走行待ちリストを、RTOSが作成する方法は、実施例17において図45で示したものと同様なので、その説明は省略する。

【0434】また、実施例18において示したように、図49に示したは、タスクの走行割合を保証するのに、必要なデータ構造を示したもので、510は“タスク開始システムクロックメモ”である。RTOSが、タスクを起動する直前に、積算システムクロックメモ503の値を記録しておく、そしてそのタスクが終了してRTOSに戻って来たときの積算システムクロックメモ503との差を求めるこによって、タスクが走った時間を求める。

【0435】また、図49に示した“基準周期”511，“タスク走行状態メモ”512“走行残りクロック数”の欄、“繰り返し回数”の欄、および“走行禁止フ

ラグ”の欄については、実施例18と同様なので、その説明を省略する。

【0436】図45に示したBB4や、BB16において、走行待ちリストの先頭ブロックの走行予定クロック数を、システムクロック数メモ501に格納した。その指定したクロック数が経過した後、RTOSはタスク起動処理を呼ぶ。

【0437】図59に示した、制御ソフトウェア実行システムA20は、稼働し始めたときなど走行割合を保証するタスクが動き始める前に、予めシステムクロック数メモに対して、基準周期511の値を設定しておく。この設定したクロック数が経過した後に、基準周期処理をRTOSが呼ぶ。基準周期処理の手順については、実施例18の図50に示した手順と同様なので、その説明を省略する。

【0438】また、走行時間警告処理の手順についても、実施例18の図51に示した手順と同様なので、その説明を省略する。

【0439】さらに、システムクロック割り込み処理の手順についても、実施例18の図52に示した手順と同様なので、その説明を省略する。

【0440】また、RTOSのスケジューラの動作についても、実施例18の図53に示した手順と同様なので、その説明を省略する。

【0441】ここで、タスク起動処理、基準時間処理、走行時間警告処理、そして、システムクロックの割り込みの処理について、TCB待ち行列への操作を中心まとめる。

【0442】タスク起動処理は、この処理によって起動するタスクのTCBを、TCB待ち行列の先頭に挿入する。また、基準時間処理は、全てのタスクを走行可能とする。走行時間警告処理は、走行時間の割合を保証しているタスクのTCBを、そうでないタスクのTCBよりも前に持っていると共に、タスク走行状態メモ512の走行禁止フラグをオンすることによって、走行時間警告処理が発生したことを、システムクロック割り込みのディスパッチ処理に対して知らせる。

【0443】システムクロック割り込みのディスパッチ処理は、走行時間警告処理によって、走行時間警告処理が発生した場合には、基準時間処理が発生するまでは、走行時間を保証しているタスクのTCBをそうでないタスクのTCBよりもTCB待ち行列の前になら。

【0444】ここで、タスク起動処理とシステムクロックのディスパッチ処理の優先順位はどちらかを選択しておく。タスク起動処理を、優先すればタスクの起動時間は正確になるが、タスクの走行割合は多少不正確になる。一方、ディスパッチ処理を優先すれば、タスクの走行割合は正確になるが、タスクの起動時間は多少不正確になる。これらは、何らかの手段によって閾値を決めておいてもよい。例えば、タスクの起動時間の遅れが、あ

る時間までは走行割合を優先するがそれ以上は、タスク起動を優先するなどの方法を取ればよい。

【0445】以上で述べたように、実施例20に係るNC装置の制御ソフトウェア実行システムA20は、通常のプライオリティーベースのスケジューリング方式しか持たないRTOSに、タスク起動周期とタスク走行割合の保証を与える制御を行える機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0446】上記実施例20によれば、NC装置の制御ソフトウェア実行システムA20は、NC装置機能を実現するタスクの正確な起動時間と走行時間による制御を行えるようになる。そのため、タスクに割り当てた時間の無駄を少なくできる。さらに、タスクをその走行時間で制御できるので、システムの構築が容易に行える。

【0447】【実施例21】次に、実施例21について説明する。実施例21に係るNC装置の制御ソフトウェア実行システムA21の全体構成は、実施例17に示したもの（図44参照）と同様であるので、その説明を省略する。

【0448】次に、実施例21に係るNC装置の制御ソフトウェア実行システムA21の動作について説明する。実施例21と同様に、各タスクは予め、あるいはそのタスクが開始させられるときにSVCによってその実行周期を登録する。

【0449】このデータは、RTOSが、例えば図2に示したようなタスク走行周期テーブル50に登録する。このタスク走行周期テーブル50における“走行周期”と“繰り返し回数”については、実施例17と同様なので、その説明を省略する。

【0450】図3において、リスト中の各走行待ちブロックは、それぞれ“タスク名”と、現在からどれだけの時間の経過後に走らなくてはならないのか“走行予定クロック数”，それに同時に走らせなくてはならないタスク名を示す“同時タスク名”から成るブロックである。

【0451】このタスク走行待ちリストを、RTOSが作成する方法は、実施例17において図45で示したものと同様なので、その説明は省略する。

【0452】また、実施例19と同様に、図55、図56は、タスクの連続走行を走行割合を保証するのに、必要なデータ構造を示したもので、図55は、“タスク終了クロック数”のリスト構造である。520は、“タスク終了クロックブロック”を表している。タスク終了ブロックの構成要素はタスク終了ブロックを連結するためのポインター、“タスク名”，“終了クロック数”などである。今までの説明と同様本実施例の説明でも内部的には全てクロック数で時間の制御を行うものとする。

“実行中フラグ”は現在連続走行を保証をされたタスクが走っているときはオン、それ以外のときはオフとす

る。このフラグは同時に、2つ以上オンすることは無い。

【0453】図56は、“タスク連続走行保証テーブル”521である。このテーブルには“タスク名”と“連続走行保証クロック数”の欄があり、連続走行保証クロック数の値はシステム起動時、またはSVCにより設定する。連続走行を指定しないタスクに対してはこの欄は0にしておく。このテーブルはタスク起動のSVCにおいて、そのパラメータとして指定することすれば、必要はなくなるが、以下の説明の都合上、ここではこのテーブルを用いることとする。

【0454】RTOSのスケジューラは、連続走行が保証されているタスクが走行しているときに、割り込みが発生しても割り込み処理の終了後の再スケジューリングは行わないことによってタスクの連続走行を保証する。連続走行が保証されているタスクの実行を終了させるのは“連続走行終了割り込み”が発生したときである。

【0455】RTOSはタスク起動のSVCが、発行されたときは、まずそのタスクが連続走行を保証されたタスクか否かを判定して、もし該当タスクなら、タスク連続走行保証テーブル（図56参照）から連続走行クロック数をシステムクロック数メモに設定する。設定したクロック数が経過すると、“連続走行終了処理”RTOSから呼ばれる。

【0456】連続走行終了処理の手順については、実施例19の図57に基づいて説明したものと同様なので、その説明を省略する。

【0457】基準周期割り込み以外の、一般的割り込み処理の手順についても、実施例19の図58に基づいて説明したのと同様なので、その説明を省略する。

【0458】スケジューラも、実施例2と同様に、同じスケジューラ（図14参照）と同じであるので、その説明を省略する。

【0459】ここで、連続走行処理と、システムクロック割り込みのディスパッチ処理について、TCB待ち行列への操作を中心にまとめる。すなわち、

① 第1に、タスク起動処理は、この処理によって起動するタスクのTCBを、TCB待ち行列の先頭に挿入する。

② 第2に、連続走行終了処理は、実行中のタスクの実行中フラグをオフして他のタスクの走行を許可する。

③ 第3に、システムクロック割り込みのディスパッチ処理は、連続走行を保証するタスクの場合は、TCB待ち行列のディスパッチを行わない。

【0460】ここで、①と③の優先順位はどちらかを選択しておく。①を優先すればタスクの起動時間は正確になるが、タスクの連続走行時間は多少不正確になる。一方、③を優先すれば、タスクの連続走行時間は正確になるが、タスクの起動時間は多少不正確になる。これらは、何らかの手段によって閾値を決めておいてよい。

例えば、タスクの起動時間の遅れが、ある時間までは連続走行を優先するがそれ以上は、タスク起動を優先するなどの方法を取ればよい。

【0461】以上で述べたように、実施例21に係るNC装置の制御ソフトウェア実行システムA21は、通常のプライオリティーベースのスケジューリング方式しか持たないRTOSに、タスク起動周期とタスクの連続走行の保証を与える制御を行える機能を追加した方式で、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0462】上記実施例21によれば、NC装置の制御ソフトウェア実行システムA21は、NC装置機能を実現するタスクの正確な起動時間と連続走行時間による制御実行することができる。そのため、タスクに割り当った時間の無駄を少なくできる。さらに、タスクをその走行時間で制御できるので、システムの構築が容易に行える。

【0463】【実施例22】次に、実施例22について説明する。図60は、実施例22に係るNC装置の制御ソフトウェア実行システムA22の全体構成を示し、その内容は、実施例17とほぼ同様である。

【0464】次に、NC装置の制御ソフトウェア実行システムA22の動作について説明する。本実施例に係る制御ソフトウェア実行システムA22のRTOSは、予め、例えば、システムを稼働したときなどに、積算システムクロック数メモ503を0に初期化しておく。これらの値は、システムクロック割り込みの毎にシステムクロック割り込み処理の中で1加算されている。

【0465】また、タスクにはいる直前に、そのクロック数メモの値を読み取って記憶しておく。また、そのタスクを抜けた直後に、再びクロック数メモの値を読み取って記憶しておいた値との差を求める。その値が、今回このタスクが走ったクロック数である。そしてタスク毎に、この値の合計を記憶しておく場所を設けておいて、そこに加えていくことにより、各タスクの走行時間の合計を記録する。

【0466】以下、本実施例に係るNC装置の制御ソフトウェア実行システムA22の動作について、さらに詳細に説明する。図61は、各タスクの走行時間を記録するのに必要なデータ構造であり、ここで、530はカウンタ値メモ、531はタスク走行時間テーブルであり、図に示す通り、タスク毎の合計クロック数の欄がある。

【0467】図62は、システムクロック割り込み処理の処理手順を示すフローチャートであり、図63は、スケジューラの処理手順を示すフローチャートである。

【0468】図62は、システムクロック割り込み処理の手順について説明したフローチャートであり、KK1は割り込みが発生したときの割り込み処理の先頭であり、最初に他の割り込みの発生を禁止する、通常この処理はメインCPU1によって自動的に行われる。KK1

6はシステムクロック数メモ501に1を加算する処理である。KK17はシステムクロック数メモ501に1を加算する処理である。

【0469】KK18はディスパッチクロック数メモ509から1減算する処理であり、KK19はその結果を0と比較する。KK20はディスパッチクロック数メモ509が0になったときの処理で、ディスパッチクロック数508で再初期化したのち、ディスパッチを行うためKK2の処理へ進む。KK19の判定結果が0でない

10 場合は直接割り込まれた処理に戻るため、KK7の処理へ移行する。

【0470】KK2は未使用のICBをICBプール(未使用ICBを保持しておく場所)から1つ獲得する。KK3はICBに、この割り込み処理(自分自身)の、実行環境を格納する。この中には次に実行する命令のアドレスとしてKK8の処理の先頭アドレスをICBに格納することも含まれる。KK4はここで作成したICBを、スケジューラが実行するICB待ち行列に追加する。KK5はKK1の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためである。

【0471】KK6は割り込まれた処理が割り込み処理またはRTOSか、それともタスクの処理中かを判定する。KK7は割り込まれた処理が割り込み処理またはRTOSを実行中のときで、このときは割り込まれた処理に直接戻る。KK8は割り込まれた処理がタスクの処理を実行中のときで、このときはまず、割り込まれたタスクのTCBを探し出す。KK9は、KK8において見つけたTCBへ割り込まれたタスクの実行環境を格納する。

【0472】KK10は、積算システムクロック数メモ503の値から、カウンタ値メモ530の値を減算する。この結果、当該タスクがこの回に連続して走ったクロック数である。KK11は、KK10で求めた値をタスク走行時間テーブル531の当該タスクの合計クロック数の欄に加える。KK12は、作成したTCBを、スケジューラが実行するTCB待ち行列に優先順位の順番に追加する。

40 【0473】KK13はスケジューラへ飛ぶ。KK14はスケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後すぐにこの処理が行われるとは限らず、他のICBが処理された後かも知れないがいずれはここへ戻ってくる。ここからが割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。KK15は割り込み処理の終了であり、通常のサブルーチンのリターン命令が多い、このリターン命令によって、スケジューラのICB処理ブロックに戻る。

【0474】図63は、本実施例におけるRTOSのス

ケジューラの動作を示したフローチャートである。説明の都合上、本実施例においては、優先順位が最低で、外部に対して何も処理を行わないアイドルタスクが常に走っているものとする。すなわち、アイドルタスクは、常に走っているものとして説明を行う。

【0475】LL1は割り込み処理（ICB）の待ち行列があるか否かを調べる処理である。LL2は割り込み処理の待ち行列が合ったときの処理でその待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行うとするとリストの整合性が保たれなくなるためである。

【0476】LL3は割り込み処理の待ち行列のリストから先頭のICBを取り除く。LL4は割り込み禁止を解除する。LL5は、LL3で取り除いたICBからKK3（図62参照）で格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して実行する。これがKK14（図62参照）の処理である。

【0477】LL6は、KK15（図62参照）から戻っているところであり、再び割り込みを禁止する。LL7は、LL3で取り除いたICBをICBプールへ返す。LL8は割り込み禁止を解除してからLL1へ戻る。後は割り込み処理待ち行列がある間はLL1からLL7の処理を繰り返す。LL9はタスク処理の待ち行列があったときの処理でその待ち行列を実行するために最初に割り込みを禁止する。

【0478】LL10は割り込み処理の待ち行列のリストから先頭のTCBを取り除く。LL11はカウンタ値メモ530にシステムクロック数メモ501の値を保存しておく。LL12は割り込み禁止を解除する。LL13はLL11でタスク処理の待ち行列（リスト）から取り除いたTCBからKK9（図62参照）で格納したタスクの実行環境を取り出して実行する。このときはその処理へ飛んでいってここへは再び戻ってこない。

【0479】以上のようにして、任意の時間、本実施例に係るシステムを走らせた後、タスク走行時間テーブル531を調べれば、各タスクが実際に走った（処理を行った）時間を知ることができる。

【0480】なお、タスクの走行時間のみではなく、タスク走行時間テーブル531に走行回数の欄を設けて、タスクの走行回数も記録するようにしておけば、以下に述べるタスクの走行周期、走行割合、走行時間を決めるときにも役立つ。

【0481】さらに、カウンタ値メモ530だけでなく、タスク終了カウンタ値メモも設けて割り込み処理の先頭で、システムクロック数メモの値をそこへ保存しておき、スケジューラのKK10（図62参照）の部分において、カウンタ値メモ530からタスク終了カウンタ値メモを減算した値を、タスクが連続して走行したクロ

ック数とすれば割り込み処理の時間も除いたタスクが純粹に走行した時間が求められる。

【0482】以上で述べたように、実施例22に係るNC装置の制御ソフトウェア実行システムA22は、タスクの実際の走行時間を知ることができる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0483】実施例22によれば、NC装置の制御ソフトウェア実行システムA22は、NC装置機能を実現するタスクの正確な実行時間を知ることにより、各タスクの走行時間の割り当てなどが、容易に行えるようになり、システムの構築が容易に行える。

【0484】【実施例23】次に、実施例23について説明する。実施例23に係るNC装置の制御ソフトウェア実行システムA23の全体構成は、実施例22と同様であるので、その説明を省略する。

【0485】次に、本実施例に係る、NC装置の制御ネットワーク実行システムA23の動作について説明する。本実施例に、制御ソフトウェア実行システムA23（図60参照）のRTOSは、予め、例えば、システムを稼働したとき等に、システムクロック数メモ501を0で初期化しておく。

【0486】また、タスクにはいる直前に、そのカウンタ値を読み取って記憶しておく。そのタスクを抜けた直後に、再びカウンタ値を読み取って記憶しておいた値との差を求める。その値が、今回このタスクが走ったクロック数である。そして、タスク毎に、この値の和を記憶しておく場所を設けておいて、そこに加えていくことにより、各タスクの走行時間の合計を記録する。

【0487】さらに、従来例におけるスケジューラのICBの処理を、スケジューラでなく、優先順位がもっと高いタスクで実行するものである。

【0488】以下、本実施例に係るNC装置の制御ソフトウェア実行システムについて、さらに詳細に説明する。図64は、スケジューラの処理手順を示すフローチャートであり、図65は、ICBを処理する最高優先順位を持った、割り込み処理実行タスクの処理を示すフローチャートである。

【0489】データ構造としては、従来例と同じICB（インタラプトコントロールブロック）を用いる。割り込み処理に関しては、従来例と同じであるので、その説明を省略する。

【0490】図64は、本実施例に係るRTOSのスケジューラの動作を説明したものであり、アイドルタスクを設けることにより、常に少なくともアイドルタスクは、実行を行っているものとする。MM1はタスク処理の待ち行列（TCB）があったときの処理でその待ち行列を実行するために最初に割り込みを禁止する。MM2は割り込み処理の待ち行列のリストから先頭のTCBを取り除く。

【0491】MM3はカウンタ値メモ530にシステムクロック数メモ501の値を保存しておく。MM4は割り込み禁止を解除する。MM5はMM3でタスク処理の待ち行列(リスト)から取り除いたTCBから割り込み処理で格納したタスクの実行環境を取り出して実行する。このときはその処理へ飛んでいってここへは再び戻ってこない。

【0492】本実施例における割り込み処理タスクの動作は、上記図26と同様である。すなわち、N1は、割り込み処理(IBC)の待ち行列があるか否かを判定する処理である。N2は、割り込み処理の待ち行列があつたと判断した場合の処理であつて、その待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行なうとするとリストの整合性が保たれなくなるためである。

【0493】また、N3は、割り込み処理の待ち行列のリストから、先頭のIBCを獲得する。N4は、割り込み禁止を解除する。N5は、N3で取り除いたIBCから、割り込み処理で格納した実行アドレスやレジスタなどの割り込み処理の実行環境を取り出して実行する。N6は、N3で取り除いたIBCをIBCプールへ返却する。その後は、N1へ戻り、割り込み処理待ち行列がある間はN1からN6の処理を繰り返す。N7は、割り込み処理(IBC)の待ち行列が無くなったときの処理でタスク終了のSVCを発行して他のタスクにメインCPU1の使用権を譲り渡す。

【0494】以上のように、任意の時間、本実施例を走らせた後、タスク走行時間テーブル531を調べれば、各タスクが実際に走った(処理を行なった)時間を知ることができる。

【0495】さらに、割り込み処理もタスクレベルで実行することにより、割り込み処理実行タスクの走行時間が、割り込み処理の時間と考えることができるため、割り込み処理の処理時間も測定することができる。また、割り込み処理実行タスク内で、割り込み要因により別の走行時間を記録できるようにしておけば、割り込み要因毎の処理時間を測定することができる。

【0496】以上に説明した通り、実施例23に係るNC装置の制御ソフトウェア実行システムA23は、タスクと割り込み処理の実際の走行時間を知ることができる機能を追加した方式であり、その方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0497】上記実施例23によれば、NC装置の制御ソフトウェア実行システムは、NC装置機能を実現するタスクと割り込み処理の正確な実行時間を知ることにより、各タスクの走行時間の割り当てなどが容易に行えるようになり、システムの構築が容易に行える。

【0498】【実施例24】次に、実施例24について

説明する。実施例24に係るNC装置の制御ソフトウェア実行システムA24の全体構成は、実施例18と同様であるので、その説明を省略する。

【0499】次に、本実施例に係るNC装置の制御ソフトウェア実行システムA24の動作について説明する。タスクや割り込み処理の走行時間を計測する手段を備え、さらにタスクの走行時間を変える何らかの手段を備えるNC装置において、それを実現するためのNC装置の制御ソフトウェア実行システムの全体構成図は、タスクの走行時間や割合の指定の仕方によって複数の構成があるが、この実施例では、説明の都合上タスクの走行割合を指定できる場合について説明する。

【0500】タスクの走行割合を保証する処理については、実施例18(図50～図51参照)とほぼ同様であり、必要なデータ構造も同じであり、タスク開始システムクロックメモ510(図49参照)、基準周期511を格納しておく領域(図49参照)、タスク走行状態メモ512(図49参照)などを備える。タスク走行状態メモ512(図49参照)は、“走行残りクロック数”、“繰り返し回数”、“走行禁止フラグ”などの欄を備える。

【0501】さらに、タスクの走行時間を測定するために、実施例22と同様に、タスク走行時間テーブル531(図61参照)を備える。また、タスク走行時間テーブル531(図61参照)にはタスク毎の合計クロック数の欄がある。なお、カウンタ値メモ530(図61参照)はタスク走行状態メモ512(図49参照)と同じ役目をするので使用しない。

【0502】また、基準時間処理の手順、走行時間警告処理の手順は、実施例18(図50、図51参照)と同様なので、その説明を省略する。

【0503】図65は、システムクロック割り込み処理の手順について説明した図である。以下の説明において、システムクロック数メモ501および走行時間警告用システムクロック数メモ505に関しては、実施例18のシステムクロック割り込み処理(図47参照)と同様に、それぞれに有効フラグを設けてその有効フラグが“有効”である場合のみシステムクロック数メモ501、走行時間警告用システムクロック数メモ505に対する処理を行なうが、その説明が煩雑になるため本実施例の説明に係るフローチャートではその処理の説明を省略する。

【0504】O01は割り込みが発生したときの割り込み処理の先頭であり、最初に、他の割り込みの発生を禁止する、通常この処理はCPUによって自動的に行われる。O020はディスパッチクロック数メモ509、システムクロック数メモ501、走行時間警告用システムクロック数メモ505のそれから1減算する処理であり、O021は積算システムクロック数メモ503に1加える処理である。

【0505】0022はskipフラグをオンする処理である。ここで、skipフラグとは、RAM4上に設けた、このシステムクロック割り込みでスケジューリングなどの処理を行うか否かを示すために用いるフラグである。0023は走行時間警告用システムクロック数メモが0になったか否かの判定であり、0になった場合は、0024のタクス起動処理（図46参照）を呼ぶ処理へ移行する。その後、0025でskipフラグをオフにする。0023の判定において、走行時間警告用システムクロック数メモ505が0でない場合には0026の処理へ移行する。

【0506】0026はシステムクロック数メモ501が0になったか否かの判定を行う処理であり、その結果、0であれば、0027で基準周期処理（図12参照）を呼び、0028でskipフラグをオフにする。

【0507】0026の判定結果が0の場合には、0029ではディスパッチクロック数メモ509が0か否かの判定を行う。その結果が0であれば、0030でskipフラグをオフにして、0031でディスパッチクロック数508の値をディスパッチクロック数メモ509へ代入することにより、ディスパッチクロック数メモ509の再初期化を実行する。

【0508】0032はskipフラグがオンかオフかの判定を行う処理で、オンの場合は、0033で割り込み禁止の解除を行った後、007へ飛び割り込まれた処理に直接復帰する。オフの場合には002へ飛び、通常のディスパッチ処理を実行する。

【0509】002は、未使用のICBをICBプール（未使用ICBを保持しておく場所）から1つ獲得する003は、ICBに、この割り込み処理（自分自身）の、実行環境を格納する。この中には次に実行する命令のアドレスとして008の処理を先頭アドレスをICBに格納することも含まれる。

【0510】004は、ここで作成したICBを、スケジューラが実行するICB待ち行列に追加する。005は001の割り込みの禁止を解除する。これ以降は、再び割り込みが発生する可能性がある。以上の処理は割り込み禁止時間をできる限り短くするためである。006は割り込まれた処理が割り込み処理またはRTOSか、それともタスクの処理中かを判定する。

【0511】007は割り込まれた処理が割り込み処理またはRTOSを実行中のときで、このときは割り込まれた処理に直接戻る。008は割り込まれた処理がタクスの処理を実行中のときで、このときはまず、割り込まれたタスクのTCBを探し出し、009で当該RCBへ割り込まれたタスクの実行環境を格納する。

【0512】0010は積算システムクロック数メモ503の値から、基準周期511に保存してある値を減算する。この結果、が当該タスクが今回連続して走ったクロック数である。これをタスク走行時間テーブル531

の当該タスクの合計クロック数の欄に加える。

【0513】0011は、ここで処理を行っているタスクが、走行する割合を保証する対象のタスクであるか否かを判定する処理である。具体的には、タスク走行状態メモ512の走行残りクロック数の欄が1以上のタスクか否かを判定する。0012は、0011で対象と判定した場合の処理で、0010で求めた値をタスク走行状態メモ512の当該タスクの走行残りクロック数の欄から減算する。

10 【0514】0013は、0010で求めた値を走行時間警告用の走行時間警告用システムクロック数メモ505に加える。走行周期を保証したタスクが、走行し終わった分だけ、走行周期を保証したタスク以外のタスクを走らせるための時間が増えるわけである。

【0515】0014は作成したTCBを、スケジューラが実行するTCB待ち行列に、優先順位の順番に追加する。ただし、このとき、タスク走行状態メモ512（図49）の走行禁止フラグの欄が1つのタスクでもオンになっていれば、TCB待ち行列において、走行禁止フラグがオフのタスクのTCBは、走行禁止フラグがオンのタスクのTCBよりも前につなぐ。

【0516】0015はスケジューラへ飛ぶ。0016はスケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後すぐにこの処理が行われるとは限らず、他のICBが処理された後かも知れないがいずれはここへ戻ってくる。ここからが割込みに固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。0017は割り込み処理の終了であり、通常のサブルーチンのリターン命令が多い、このリターン命令によってスケジューラのICB処理部に戻る。

【0517】図66は、スケジューラの動作を示すフローチャートである。本実施例においては、優先順位が最低で、外部に対して何も処理を行わないアイドルタスクが常に走っているものとする。

【0518】図66において、PP1は割り込み処理（ICB）の待ち行列が有るか否かを調べる処理であり、PP2は割り込み処理の待ち行列があったときの処理でその待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行おうとするとリストの整合性が保たれなくなるためである。

【0519】PP3は割り込み処理の待ち行列のリストから先頭のICBを取り除く。PP4は割り込み禁止を解除する。PP5はPP3で取り除いたICBから003（図65参照）で格納した実行アドレスやレジスターなどの割り込み処理の実行環境を取り出して実行する。

【0520】PP6は0017（図65参照）から戻ってくるところであり、再び割り込みを禁止する。PP7はPP3で取り除いたICBをICBプールへ返す。P

P 8は割り込み禁止を解除してからP P 1へ戻る。後は割り込み処理待ち行列がある間はP P 1からP P 7の処理を繰り返す。

【0521】P P 9はタスク処理の待ち行列を実行するために、割り込みを禁止する。P P 10は割り込み処理の待ち行列のリストから先頭のT C Bを獲得する。P P 11はタスク開始システムクロックメモ510にタスク走行時間計測用の積算システムクロック数メモ503の値を保存しておく。

【0522】P P 12は割り込み禁止を解除する。P P 13はP P 10でタスク処理の待ち行列（リスト）の先頭から、獲得したT C BからO O 9（図65参照）が格納したタスクの実行環境を取り出して実行する。このときはその処理へ飛んでいって、ここへは再び戻ってこない。

【0523】以上において説明したように、実施例24に係るNC装置の制御ソフトウェア実行システムA24はタスクの走行割合を設定でき、また、タスクの割り込み処理の実際の走行時間を知ることができる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【0524】実施例24によれば、NC装置の制御ソフトウェア実行システムA24は、NC装置機能を実現するタスクと、割り込み処理の正確な実行時間を知ることにより、各タスクの走行時間の割り当てが、容易に行えるようになりシステムの構築が容易となる。

【0525】【実施例25】次に、実施例25について説明する。実施例25に係るNC装置の制御ソフトウェア実行システムA25の全体構成は、実施例18（図48参照）とほぼ同様であるので、その説明を省略する。

【0526】次に、NC装置の制御ソフトウェア実行システムA25の動作について説明する。NC装置の運転モード選択部分の処理手順は図29に示したものと同様である。すなわち、Q1は、NC装置のモードがグラフィックチェックか否かを判定する部分である。Q2は、グラフィックチェックモードであると判断した場合であり、その場合はグラフィック表示タスクの割り当てを増やし、他の、例えば、自動モード処理タスクの割り当てをその分減らす。

【0527】Q3は、グラフィックチェックモードではないと判断した場合で、自動モードか否かを判定する処理である。Q4は、自動モードであると判断した場合であって、自動モード処理タスクの割り当てを増やし、他の処理タスクの割り当てを減らす。Q5は、手動モードか否かのチェックである。Q6は、手動モードであると判断した場合であって、手動処理タスクの割り当てを増やし、他の処理タスクの割り当てを減らす。それ以降は、その他のモードの場合であるが、その説明は省略する。

【0528】以上において、説明したとおり、実施例2

5に係るNC装置の制御ソフトウェア実行システムA25は、NC装置の運転モードによってタスクの走行割合を変化させて、NC装置の制御ソフトウェアを実行するものである。

【0529】実施例25によれば、NC装置の制御ソフトウェア実行システムA25は、NC装置の運転モードによってタスクの走行割合を変化させることにより、モード毎にタスクに適した時間だけ、タスクを走らせることができ、NC装置の実行速度が向上する。

10 【0530】【実施例26】次に、実施例26について説明する。実施例26に係るNC装置の制御ソフトウェア実行システムA26は、同一の加工プログラムを複数回走らせるとき（同じ加工を複数回行うとき）、初回加工時における各タスクの走行状況に基づいて2回目以降の加工時における各タスクの走行配分を最適化するものである。

【0531】実施例26に係るNC装置の制御ソフトウェア実行システムA26の全体構成は、上記実施例18（図48参照）とほぼ同様であるので、その説明を省略

20 するが、以下の説明の都合上、本実施例においては、例えば、実施例24と同様に、タスク毎の走行割合を指定する手段と、実際のタスクの走行状況を記録できる手段を備えているものとする。

【0532】使用するデータ構造としては、実施例24と同様、タスク開始システムクロックメモ510（図49参照）、基準周期511を格納しておく領域（図49参照）、タスク走行状態メモ512（図49参照）、タスク走行時間テーブル531（図61参照）などを備える。タスク走行状態メモ512（図49参照）は、“走行残りクロック数”，“繰り返し回数”，“走行禁止フラグ”などの欄があり、タスク走行時間テーブル531（図61参照）にはタスク毎の“合計クロック数”的欄がある。

【0533】基準時間割り込み処理の手順、走行時間警告割り込み処理の手順は、実施例18（図50、図51参照）と同様であり、システムクロック割り込み処理の手順は、実施例24（図66参照）と同様なので、その説明は省略する。

【0534】複数回加工時の処理を時間を短縮するための、各タスクの走行割当時間の最適化するための手順は図30のフローチャートと同様である。すなわち、R1は同一加工の1回目か、2回目以降かを判定する処理である。R2は1回目の加工時の処理で、各タスクの総走行時間（クロック数）を計測し、記録する。R3は走行したタスクの内、アイドルタスクなどの必要のないタスクを除いた他の必要なタスクの走行時間の合計を計算する。

【0535】R4は、R3で求めた合計に基づいて、必要なタスクの走行割合を計算する。走る必要がないタスクの走行割合は0とする。R5は、R4で計算した割合

をタスク走行状態メモ 512 (図49参照) の走行割合の欄に登録する。R6は2回目以降の加工時の処理で、R5で登録したタスク走行状態メモ 512 (図49参照) の各タスクの走行割合により各タスクを走行させる。

【0536】以上説明した通り、実施例26に係るNC装置の制御ソフトウェア実行システムA26は、1回目の加工によって最適なタスクの走行割合を自動的に設定する機能を備えてNC装置の制御ソフトウェアを実行するものである。

【0537】実施例26によれば、NC装置の制御ソフトウェア実行システムA26は、機能を実現する各タスクの走行時間の割り当てを自動的に最適なものにすることにより、繰り返し加工における加工時間の短縮を実行できる。

【0538】【実施例27】次に、実施例27について説明する。実施例27に係るNC装置の制御ソフトウェア実行システムA27は、予め登録された、各タスクの基準の走行割合や、全走行時間と、実際に走っているNC装置の制御ソフトウェア実行システムの各タスクのデータが、予め指定した割合以上にずれている場合は、タスクの異常と判定する。

【0539】実施例27に係るNC装置の制御ソフトウェア実行システムA27の全体構成は、実施例24 (図48参照) と同様であるので、その説明は省略する。また、実際に走っているNC装置の制御ソフトウェア実行システムの各タスクのデータを測定するのは、実施例と同様であるので、その説明は省略する。

【0540】各タスクが正常に走っているか否かを判定するのに必要なデータ構造を、図67に示す。550は“タスク走行標準データ”で、タスク毎に“走行割合”や“全走行クロック数”などの欄がある。それらのデータは、システム生成時に予め作成しておくか、あるいは何らかのSVCによって、そのデータを使うよりも前の、任意の時期に設定しておいてもよい。

【0541】図において、551は、この割合まではタスク走行時間のずれがあっても、タスク異常と見做さない許容度を格納する“走行時間判定許容度”である。552は、この範囲内であれば、タスク走行割合にずれがあってもタスク異常と見なさない許容度を格納する“走行割合判定許容度”である。この走行割合判定許容度には下限欄と上限欄があり、それがこの範囲内ならばタスク異常と見なさないものである。

【0542】各タスクが正常に走っているか否かを判定するための、タスク走行時間テーブル531 (図61参照) とタスク走行標準データ550 (図67参照) の比較は、例えば、タスクが切り換えるときに、スケジューラで行ってもよいし、あるいは、定期的に走るタスク異常判定タスクを設けて、そのタスクにより判定を行ってもよい。以下の説明では、タスクが正常に走っているか

否かの判定を、異常判定タスクで行うものとする。また、走行時間判定許容度551は全走行時間と、走行割合について別々に設けてもよいが、ここでは説明の都合上、同じ値を用いるものとする。

【0543】図68は、異常判定タスクの動作を示すフローチャートである。SS1は、異常判定タスクの先頭で、タスク走行時間テーブル531 (図61参照) の、タスク毎の合計クロック数の欄の値の全ての和を求める処理である。SS2は、全てのタスクに対して、以下の

10 SS3からSS5までの処理を行ったか否かの判定であり、全タスクに対して処理を終了すればSS6へ処理が移行する。

【0544】SS3は、タスク走行時間テーブル531 (図61参照) の、当該タスクの合計クロック数の値と、タスク走行標準データ550の全走行クロック数を比較する。当該タスクのタスク走行時間テーブル531 (図61参照) の合計クロック数の値が、タスク走行標準データ550の全走行クロック数の値に走行時間判定許容度551の値を乗じたものよりも小さいか等しければ、このタスクは正常であると見なす。

【0545】SS4は、SS3において、タスクに異常が発生したと判定したときの処理で、例えば、オペレーターにタスク異常が発生したことを知らせるメッセージを送るなどの、何れかのエラー処理を行う。

【0546】SS5は、タスクの走行割合のチェックである。タスク走行時間テーブル531 (図61参照) の合計クロック数の値を、全タスク分加えて、それで当該タスクの合計クロックを割ることにより当該タスクの走行割合を求める。その値がタスク走行標準データ550の走行割合判定許容度の下限と上限の間にあれば、このタスクは正常であると見なす。もし、異常があると判定したときはSS4に処理が移行する。SS6は、全タスクを判定した結果、異常状態のタスクは見つからなかつたので、タスク終了のSVCを発行し、次に呼ばれるまで何もしない。

【0547】以上で説明した通り、実施例27に係るNC装置の制御ソフトウェア実行システムA27にあっては、NC装置の機能を実現しているタスクに以上が発生したときにそれを検知する機能を備えている。

【0548】実施例27によれば、NC装置の制御ソフトウェア実行システムA27は、NC装置の機能を実現しているタスクに異常が発生したときにそれを検知する機能を備えているため、NC装置に異常が発生したことが直ちに判定することができる。

【0549】【実施例28】次に、実施例28について説明する。実施例28に係るNC装置の制御ソフトウェア実行システムA28の全体構成は、実施例18 (図48参照) と同じであるので、その説明は省略する。

【0550】ただし、本実施例では、RAM4上のデータとしてはタスク走行時間計測用の積算システムクロッ

クメモ503, タスク停止用に使用するシステムクロック数メモ501, ディスパッチクロック数508, ディスパッチクロック数メモ509を使用する。

【0551】図69は、本実施例において、必要なデータ構造を示したものであり、600は“タスク停止テーブル”である。各タスクは予め、あるいはそのタスクが開始させられるときにSVCによって、タスクが起動してから停止するまでのクロック数をこのタスク停止テーブル600の“停止クロック数”の欄に登録しておく。指定しないタスクはこの値を0としておく。

【0552】510(図49参照)は“タスク開始システムクロックメモ”である。RTOSが、タスクを起動する直前に、積算システムクロック数メモ503(図48参照)の値を記録しておく。そして、そのタスクが終了してRTOSに戻って来たときの、積算システムクロック数メモ503との差を求めてタスク走行時間を求める。611は“タスク停止クロックメモ”である。タスク停止処理をすぐに処理できないときは、このメモをオンしておいて後で処理する。

【0553】本実施例に係る、NC装置の制御ソフトウェア実行システムA28は、タスクを起動するたびに、指定されればタスク停止テーブル600の停止クロック数をシステムクロック数メモ501に設定する。そして、毎回のシステムクロック割り込み時に、RTOSがシステムクロック数メモ501を減じていって0になったところで、タスク停止処理を呼んで当該タスクを強制的に停止させる。

【0554】もし、タスク停止処理を呼び前に、他のタスクによってCPUを横取りされた場合は、積算システムクロックメモ503(図48参照)と、タスク停止クロックメモ611から、当該タスクの走行クロックを求め、その分をタスク停止テーブル600の停止クロック数から減ずる。このようにして、指定タスクの最大走行クロック数を制限することができる。

【0555】次に、NC装置の制御ネットワーク実行システムA28の動作について説明する。最初にタスク停止処理について説明する。図70は、タスク停止処理の手順について説明したフローチャートである。

【0556】WW2は割り込まれた処理がタスク処理中か、それ以外かを判定する処理部分である。WW3はタスク処理中ではなかったときで、その場合には、タスク停止クロックメモ611をオンする。WW4は割り込み禁止を解除して、割り込まれた処理へ戻る。WW5はタスク処理中に割り込みが発生した場合で、まず割り込まれたタスクのTCBを探し出す。

【0557】また、WW6は、WW4で見つけたTCBへ割り込まれたタスクの実行環境を格納する。WW7はタスク停止SVCと同様の当該タスクの終了処理を行う。例えば、当該TCBのステータスの欄を停止にして、TCB待ち行列から、停止中タスクの待ち行列へつ

なぎ変える等の処理である。

【0558】図71は、システムクロック割り込み処理の手順を示すフローチャートである。以下の説明において、システムクロック数メモ501に関しては、実施例18のシステムクロック割り込み処理(図47参照)と同様に、有効フラグを設けてその有効フラグが“有効”である場合のみシステムクロック数メモ501に対する処理を行えばよいが、説明が煩雑になるため、本実施例の説明のフローチャートではその処理を省略する。

10 【0559】XX1は、割り込みが発生した時の割り込み処理の先頭であり、最初の他の割り込みの発生を禁止する、通常この処理はCPUによって自動的に行われる。

【0560】XX20は、ディスパッチクロック数メモ509、システムクロック数メモ501のそれぞれから1減算する処理であり、XX21は積算システムクロック数メモ503に1加える処理であり、XX22はskipフラグをオンする処理である。ここで、skipフラグは、RAM4上に設けた、このシステムクロック割り込みでスケジューリングなどの処理を行うか否かを示すために用いるフラグである。

20 【0561】XX23は、システムクロック数メモ501が、0になったか否かの判定を行う処理であり、その結果が0と判断した場合は、XX24で、図41に示したタスク停止処理を呼ぶ。その後、XX25でskipフラグをオフにする。ここで、skipフラグとはRAM4上にあり、ディスパッチ処理を呼ぶか否かの判定に使用するフラグである。

【0562】XX23の結果が0でないと判断したときは、XX26で、ディスパッチクロック数メモ509が0か否かの判定を行う。その結果が、0であれば、XX27でskipフラグをオフにして、XX28で、ディスパッチクロック数508の値をディスパッチクロック数メモ509へ代入することにより、ディスパッチクロック数メモ509の再初期化を実行する。

【0563】XX29は、skipフラグがオンかオフかの判定を行う処理で、オンの場合にはXX30で割り込み禁止の解除を行った後、XX7へ飛び、割り込まれた処理に直接復帰する。オフの場合にはXX2へ飛び、通常のディスパッチ処理を行う。

【0564】XX2は、未使用的ICBをICBプール(未使用ICBを保持しておく場所)から1つ獲得する。XX3は、ICBに、この割り込み処理(自分自身の)の、実行環境を格納する。この中には次に実行する命令のアドレスとしてXX8の処理の先頭アドレスをICBに格納することも含まれる。XX4は、ここで、作成したICBを、スケジューラが実行するICB待ち行列に追加する。XX5は、XX1の割り込みの禁止を解除する。これ以降は再び割り込みが発生する可能性がある。異常の処理は割り込み禁止時間をできる限り短くす

るためである。

【0565】XX6は、割り込まれた処理が、割り込み処理またはRTOSか、それともタスクの処理中かを判定する。XX7は、割り込まれた処理が割り込み処理またはRTOSを実行中のときで、このときは割り込まれた処理に直接戻る。XX8は、割り込まれた処理がタスクの処理を実行中のときで、このときはまず割り込まれたタスクのTCBを探し出す。XX9は、XX8で見つけたTCBへ、割り込まれたタスクの実行環境を格納する。

【0566】XX10は、タスク停止クロックメモ611がオンか否かを調べる処理である。XX11は、タスク停止クロックメモ611がオンのときの処理で、タスク停止SVCと同様の、当該タスクの終了処理を行う。例えば、当該TCBのステータスの欄を停止にしてTCB待ち行列から、停止中タスクの待ち行列へつなぎ変える等の処理である。

【0567】XX12は、タスク停止クロックメモ611がオフのときの処理で、積算システムクロック数メモ503と、タスク開始クロックメモの差から当該タスクの走行クロック数を求め、タスク停止テーブル600の停止クロック数から減ずる。XX13は、XX9において作成したTCBを、スケジューラが実行するTCB待ち行列にプライオリティーの順に追加する。XX14は、スケジューラへ飛ぶ。

【0568】XX15は、スケジューラから戻ってくるとここへ来る処理である。スケジューラへ飛んだ後すぐにこの処理が行われるとは限らず、他のICBが処理された後かもしれないが、いずれはここへ戻ってくる。ここから割込み毎に固有の本来の割り込み処理であり、割り込み要因によって処理は異なる。XX16は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が多い、このリターン命令によって、スケジューラのICB処理部に戻る。

【0569】図72を用いて、本実施例に係るNC装置の制御ソフトウェア実行システムA28のスケジューラの動作を説明する。以下の、説明の都合上、本実施例においては、優先順位が最低で、外部に対して何も処理を行わないアドレスタスクが常に走っているものとする。

【0570】YY1は、割り込み処理(ICB)の待ち行列があるか否かを調べる処理である。YY2は、割り込み処理の待ち行列があったときの処理でその待ち行列を実行するために最初に割り込みを禁止する。これは、割り込み処理の待ち行列のリストを処理している最中に、割り込みが発生して割り込み処理の待ち行列に追加を行おうとするリストの整合性が保たれなくなるためである。

【0571】YY3は割り込み処理の待ち行列のリストから先頭のICBを取り除く。YY4は割り込み禁止を解除する。

【0572】YY5は、YY3で取り除いたICBからXX3で格納した実行アドレスやレジスターなどの割り込み処理の実行環境を取り出して実行する。これがXX15(図71参照)である。YY6は、XX16から戻ってくるところであり、再び割り込みを禁止する。YY7は、YY3で取り除いたICBをICBプールへ返す。YY8は、割り込み禁止を解除してからYY1へ戻る。後は割り込み処理待ち行列がある間はYY1からYY7の処理を繰り返す。

10 【0573】YY9は割り込みを禁止する。YY10は割り込み処理の待ち行列のリストから先頭のTCBを獲得する。YY11は当該TCBのタスクが、タスク停止テーブル600の停止クロック数が0以上のタスクである場合は、当該タスクのタスク停止テーブル600の停止クロック数の値をシステムクロック数メモ501に代入する。

【0574】この設定した数だけのクロック数が経過すると、図71で説明したシステムクロック割り込みがタスク停止処理を呼ぶ。YY12は、割り込み禁止を解除する。YY13は、YY10でタスク処理の待ち行列(リスト)から獲得したTCBからXX9(図71)で格納したタスクの実行環境を取り出して実行する、そしてその処理へ飛んで行き、ここへ戻ってこない。

【0575】以上、説明したように、実施例28に係るNC装置の制御ネットワーク実行システムA28は、通常のプライオリティーベースのスケジューリング方式しか持たないRTOSに、タスク走行時間の制限を行える機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

30 【0576】実施例28によれば、NC装置の制御ソフトウェア実行システムは、NC装置機能を実現するタスクの、走行時間による制限を行えるようになる。そのため、タスクに割り当てた時間を超えてタスクが走ることがなくなり、システムの構築が容易となる。

【0577】【実施例29】次に、実施例29について説明する。実施例29に係るNC装置の制御ソフトウェア実行システムは、加工プログラム解析処理タスクなどを、従来例のように、1ブロック処理する毎にタスクを抜けるのではなく、加工プログラムが存在する限り、解析を続けるようにして作成しておき、解析の中断は、時間によるものとする。

【0578】実施例29に係るNC装置の制御ソフトウェア実行システムA29の、加工プログラム解析タスクの実行手順を図43に基づいて説明する。Z1は、加工プログラム解析タスクの先頭であり、指定された加工プログラムのブロックを全て解析し終えたかを判定する。

【0579】Z2は、全ての加工プログラムのブロックの解析を終えた場合の処理で、タスク終了SVCを発行して、再び起動させるまで何もしない。Z3は、未解析50 の加工プログラムのブロックが存在する場合の処理で、

ブロックの解析を実行する。以後、全ブロックの解析が終了するまでZ1の処理を繰り返す。

【0580】以上で説明したように、実施例29に係るNC装置の制御ソフトウェア実行システムA29は、加工プログラム解析タスクを、プログラムがある限り、解析し続けるようにして、解析の中断を時間によって行う、NC装置の制御ソフトウェアを実行するものである。

【0581】上記実施例29によれば、NC装置の制御ソフトウェア実行システムA29は、加工プログラムの解析を行うタスクの走行、中断を時間によって行うようにしたので、加工プログラムのブロックの大きさによる、処理時間の余りが無くなり、特に微小線分からなる加工プログラムの解析において処理能力の向上が期待できる。さらに、タスクの作成方法として、加工プログラムのブロック毎に、終了するなどの煩雑な処理が不要となる。

【0582】(各実施例の効果) 以上のように、この発明の各実施例に係るNC装置の制御ネットワーク実行システムによれば、NC装置の機能を分担して実現している、タスクの制御をきめ細かく実行でき、各タスクの処理時間を最適化することができるため、無駄時間を少なくて、全体の処理時間を早めることができる。また、NC装置の制御ネットワーク実行システムの異常を、システム自体が判定できる手段を備えているため、異常事が発生したときの回避処理も容易となる。

### 【0583】

【発明の効果】この発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの時間による制御を小さな単位で行なえるようになり、さらにタスクを起動する周期を簡単に指定できるため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御でき、システムの構築が容易になる。

【0584】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの走行時間による制御を実行することができるため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

【0585】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクに連続した走行時間を保証できるようになるため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

【0586】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な起動時間と走行時間による制御を行なえるようになるため、タスクに割り当てた時間の無駄

を少なくでき、さらに、タスクをその走行時間で制御できるため、システムの構築が容易になる。

【0587】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な起動時間と連続走行時間による制御を実行することができるため、タスクに割り当てた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

10 【0588】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な実行時間を知ることができたため、各タスクの走行時間の割り当てなどが、容易に行なえるようになり、システムの構築が容易になる。

【0589】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクと割り込み処理の正確な実行時間を知ることができたため、各タスクの走行時間の割り当てなどが、容易に実行できるようになり、システムの構築が容易になる。

20 【0590】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、全てのタスクの走行時間を記録でき、さらに全てのタスクの走行時間の割合を指定できるようになるため、各タスクへの走行時間の割り当てを最適なものになる。したがって、割り込み処理の正確な実行時間を知ることができ、各タスクの走行時間の割り当てなどが、容易に行なえるようになり、システムの構築が容易になる。

【0591】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、装置の運転状態により、各タスクへの走行時間の割り当てを自動的に最適なものにして、各タスクに無駄時間がなくなる、すなわち、NC装置の運転モードによってタスクの走行割合を変化させることにより、モード毎にタスクに適した時間だけタスクを走らせることができ、NC装置の実行速度が向上する。

30 【0592】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置を運転しながら、各タスクに割り当てる走行時間の割合を変化させたため、各タスクに無駄時間がなくなり、機能を実現する各タスクへの走行時間の割り当てを自動的に最適なものにすることにより、繰り返し加工における加工時間の短縮が実現する。

40 【0593】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、走行時間の比較により異常が発生したタスクを特定できる、すなわち、NC装置の機能を実現しているタスクに異常が発生したとき、それを検知することができるため、NC装置に異常が発生したことを直ちに判定することができる。

50 【0594】また、次の発明に係る制御ソフトウェア実

行システムの制御方法にあっては、NC装置の自動加工において、エラーが発生したと判定したときは、工具を変えて再度自動加工を実行するため、エラーの発生によって加工処理が止まる可能性が低くなり、自動的に連続加工が実現し、NC装置の連続運転の信頼性が向上する。

【0595】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置の自動加工において、エラーが発生したと判定したときは、工具速度を変えて再度自動加工を行なうため、エラーの発生により加工が止まる可能性が低くなり、自動的に連続加工ができる、NC装置の連続運転の信頼性が向上する。

【0596】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置の自動加工によって、エラーが発生したと判定したときは、別の加工プログラムで、再度自動加工を行なうため、エラーの発生により加工が止まる可能性が低くなり、自動的に連続加工ができるNC装置の連続運転の信頼性が向上する。

【0597】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの、走行時間による制限を行なうようするため、タスクに割り当てた時間を越えて、タスクが走ることがなくなり、システムの構築が容易になる。

【0598】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置の機能を、分担して実現しているタスクの制御をきめ細かく行なえるので、各タスクの処理時間を最適化することができるため、無駄時間を少なくて、全体の処理時間を短縮することができる。また、NC装置の制御ソフトウェア実行システムの異常をシステム自体が判定できるため、異常状態が発生したときの回避処理が容易となる。

【0599】また、この発明に係るNC装置の制御ソフトウェア実行システムの制御方法は、NC装置の機能を実現するタスクの時間による制御を小さな時間大気で行えるようになり、さらにタスクを起動する周期を簡単に指定できるため、タスクに割り当てた時間の無駄を少なくてでき、さらに、タスクをその走行時間で制御できるため、システムの構築が容易になる。

【0600】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、NC装置機能を実現するタスクを走行時間割合による制御を実行することができるため、タスクにわりしてた時間の無駄を少なくてでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

【0601】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、NC装置機能を実現するタスクに連続した走行時間を保証できるようになるため、タスクに割り当てた時間の無駄を少なくてでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

【0602】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な起動時間と、走行時間による制御を行えるようになるため、タスクに割り当てた時間の無駄を少なくてでき、さらに、タスクをその走行時間で制御できるため、システムの構築が容易になる。

【0603】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な起動時間と連続走行時間による制御を実行することができるため、タスクに割り当てた時間の無駄を少なくてでき、さらに、タスクをその走行時間において制御できるため、システムの構築が初になる。

【0604】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を実現するタスクの正確な実行時間をしてることができるため、各タスクの走行時間割り当てなどが、容易に行えるようになり、システムの構築が容易になる。

【0605】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を、実現するタスクと割り込み時間の正確な実行時間を知ることができるために、各タスクの走行時間の割り当てなどが、容易に実行できるようになり、システムの構築が容易になる。

【0606】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、全てのタスクの走行時間を記録でき、さらにタスクの走行時間の割合を指定できるため、各タスクへの走行時間の割り当てを最適なものにできる。さらに、割り込み処理の正確な実行時間も知ることができ、各タスクへの走行時間の割り当てなどが、容易に行えるようになり、システムの構築が容易になる。

【0607】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、装置の運転状態により、各タスクへの走行時間の割当を自動的に最適なものにして、各タスクに無駄時間なくなる。すなわち、NC装置の運転モードによって、タスクの走行割合を変化させることにより、モード毎にタスクに適した時間だけタスクを走らせることができNC装置の実行速度が向上する。

【0608】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置を運転しながら、各タスクに割り当てる走行時間の割合を変換させるため、各タスクに無駄時間がなくなり、機能を実現する各タスクへの走行時間の割当を自動的に最適なものとすることにより、繰り返し加工における加工時間の短縮が実現できる。

【0609】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、走行時間の比較により、異常が発生したタスクを特定できる。すなわち、NCの機能を実現しているタスクに異常が発生した時、そ

れを検知することができるため、NC装置に異常が発生したことを直ちに判定することができる。

【0610】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を、実現するタスクの、走行時間による制限を行うようするため、タスクに割り当てた時間を超えて、タスクが走ることがなくなり、システムの構築が容易になる。

【0611】また、次の発明に係る制御ソフトウェア実行システムの制御方法にあっては、NC装置機能を、分担して実現しているタスクの制限をきめ細かく行えるので、各タスクの処理時間を最適化することができるため、無駄時間を少なくして、全体の処理時間を短縮することができる。また、NC装置の制御ソフトウェア実行システムの異常をシステム自体が判定できるため、異常事態が発生したときに、回避処理が容易となる。

【図面の簡単な説明】

【図1】 実施例1に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図2】 タスク走行周期テーブルの構造を示す説明図である。

【図3】 タスク走行待ちリストの構造を示す説明図である。

【図4】 実施例1に係るNC装置の制御ソフトウェア実行システムがタスク走行待ちリストを作成するときの処理手順を示すフローチャートである。

【図5】 走行待ちリストの構造を示す説明図である。

【図6】 走行待ちリストに、ブロックを追加した後の状態を示す説明図である。

【図7】 タスク起動割り込みの処理手順を示すフローチャートである。

【図8】 一般の割り込みの処理手順を示すフローチャートである。

【図9】 実施例2に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図10】 実施例2に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図11】 基準割り込みの処理手順を示すフローチャートである。

【図12】 走行時間警告割り込みの処理手順を示すフローチャートである。

【図13】 一般の割り込みの処理手順を示すフローチャートである。

【図14】 スケジューラの処理手順を示すフローチャートである。

【図15】 実施例3に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図16】 タスク終了チック数リストの構造を示す説明図である。

【図17】 タスク連続走行保証テーブルの構造を示す

説明図である。

【図18】 連続走行割り込みの処理手順を示すフローチャートである。

【図19】 一般の割り込みの処理手順を示すフローチャートである。

【図20】 実施例4に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図21】 実施例6に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図22】 実施例6に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図23】 一般の割り込みの処理手順を示すフローチャートである。

【図24】 スケジューラの処理手順を示すフローチャートである。

【図25】 スケジューラの処理手順を示すフローチャートである。

【図26】 割り込み処理タスクの処理手順を示すフローチャートである。

【図27】 一般の割り込みの処理手順を示すフローチャートである。

【図28】 スケジューラの処理手順を示すフローチャートである。

【図29】 運転モード選択処理の処理手順を示すフローチャートである。

【図30】 タスク走行時間の割り当てを最適化するための処理手順を示すフローチャートである。

【図31】 実施例11に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図32】 異常判定タスクの処理手順を示すフローチャートである。

【図33】 実施例12に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図34】 工具選択処理の処理手順を示すフローチャートである。

【図35】 実施例13に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図36】 速度再計算処理の処理手順を示すフローチャートである。

【図37】 実施例14に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図38】 別加工プログラムを選択する処理の処理手順を示すフローチャートである。

【図39】 実施例15に係るNC装置の制御ソフトウェア実行システムにおける必要なデータ構造を示す説明

図である。

【図 4 0】 タスク停止割り込み処理の処理手順を示すフローチャートである。

【図 4 1】 一般の割り込み処理の処理手順を示すフローチャートである。

【図 4 2】 スケジューラの処理手順を示すスローチャートである。

【図 4 3】 加工プログラム解析タスクの処理手順を示すフローチャートである。

【図 4 4】 実施例 17 (21) に係る NC 装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図 4 5】 実施例 17 に係る NC 装置の制御ネットワーク実行システムがタスク走行待ちリストを作成するときの処理手順を示すフローチャートである。

【図 4 6】 タスク起動の処理手順を示すフローチャートである。

【図 4 7】 システムクロック割り込みの処理手順を示すフローチャートである。

【図 4 8】 実施例 18 (24, 25, 26, 27, 28) に係る NC 装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図 4 9】 実施例 18 に係る NC 装置の制御ネットワーク実行システムにおいて必要なデータ構造を示す説明図である。

【図 5 0】 基準周期リセット処理の手順を示すフローチャートである。

【図 5 1】 走行時間警告処理の手順を示すフローチャートである。

【図 5 2】 システムクロック割り込み処理の手順を示すフローチャートである。

【図 5 3】 スケジューラの処理手順を示すフローチャートである。

【図 5 4】 実施例 19 に係る NC 装置の制御ネットワーク実行システムの全体構成を示すブロック図である。

【図 5 5】 タスク終了クロック数リストの構造を示す説明図である。

【図 5 6】 タスク連続走行保証テーブルの構造を示す説明図である。

【図 5 7】 タスク連続走行終了処理の手順を示すフローチャートである。

【図 5 8】 システムクロック割り込み処理の手順を示すフローチャートである。

【図 5 9】 実施例 20 に係る NC 装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図 6 0】 実施例 22 (23) に係る NC 装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図 6 1】 実施例 22 に係る NC 装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明

図である。

【図 6 2】 システムクロック割り込み処理の手順を示すフローチャートである。

【図 6 3】 スケジューラの処理手順を示すフローチャートである。

【図 6 4】 スケジューラの処理手順を示すフローチャートである。

【図 6 5】 システムクロック割り込み処理の手順を示すフローチャートである。

【図 6 6】 スケジューラの処理手順を示すフローチャートである。

【図 6 7】 実施例 27 に係る NC 装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【図 6 8】 異常判定タスクの処理手順を示すフローチャートである。

【図 6 9】 実施例 28 に係る NC 装置の制御ソフトウェア実行システムにおける必要なデータ構造を示す説明図である。

20 【図 7 0】 タスク停止処理の処理手順を示すフローチャートである。

【図 7 1】 システムクロック割り込み処理の手順を示すフローチャートである。

【図 7 2】 スケジューラの処理手順を示すフローチャートである。

【図 7 3】 従来における NC 装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【図 7 4】 NC 装置の制御ソフトウェア実行システムの、加工プログラムの解析実行手順を示す説明図である。

【図 7 5】 割り込み処理を示す概念図である。

【図 7 6】 各タスク動作の時間的関係を示すタイミングチャートである。

【図 7 7】 各タスク動作の時間的関係を示すタイミングチャートである。

【図 7 8】 自動プログラム動作の状態を示す説明図である。

【図 7 9】 解析タスクの処理の処理手順を示すフローチャートである。

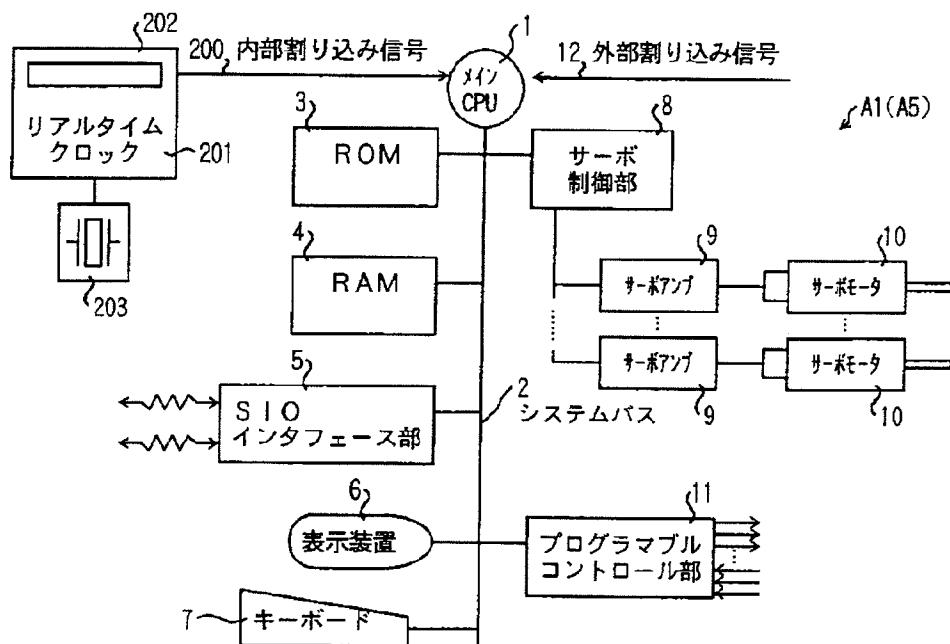
【符号の説明】

1 メイン C P U, 2 システムバス, 3 R O M, 4 R A M, 5 S I O インタフェース, 6 表示装置, 7 キーボード, 8 サーボ制御部, 9 サーボアンプ, 10 サーボモータ, 12 外部割り込み信号, 2 0 0 内部割り込み信号, 2 0 1 リアルタイムクロック (プログラマブルインタラプトコントローラ), 2 0 2 内部カウンタレジスタ, 2 0 3 水晶発信器, 2 0 4 ～ 2 0 7 内部カウンタレジスタ, 2 1 0 タスク開始チックメモ, 2 1 1 基準周期, 2 1 2 タスク走行状態メモ, 2 2 0 タスク終了チェック数リスト, 2 3 0

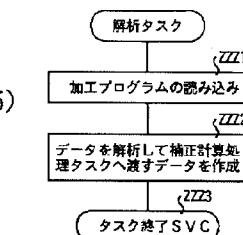
カウンタ値メモ, 231 タスク走行テーブル, 250  
 タスク走行標準テーブル, 251 走行時間判定許容度, 252 走行割合判定許容度, 360 工具選択用フラグ, 362 工具選択処理スキップフラグ, 363 選択工具メモ, 370 工具速度決定フラグ, 371 工具速度計算処理スキップフラグ, 372 工具速度メモ, 373 工具速度変更割合, 380 加工プログラムメモ, 381 加工プログラムスケジューリングテーブル, 382 加工不能プログラムメモ, 400 タスク停止テーブル, 411 タスク停止割り込みメモ, 500 システムクロック, 501 システムクロック数メモ, 502 システムクロック数メモ有効フラグ, 503 積算システムクロック数メモ, 505 走\*

\* 行時間警告用システムクロック数メモ, 506 タスク連続走行終了クロック数メモ, 507 タスク連続走行終了クロック数メモ有効フラグ, 508 ディスパッチクロック数, 509 ディスパッチクロック数メモ, 510 タスク開始システムクロックメモ, 511 基準周期, 512 タスク走行状態メモ, 520 タスク終了クロックブロック, 521 タスク連続走行保証テーブル, 530 カウンタ値メモ, 531 タスク走行時間テーブル, 550 タスク走行標準データ, 551 10 走行時間判定許容度, 552 走行割合判定許容度, 600 タスク停止テーブル, 611 タスク停止クロックメモ

【図1】



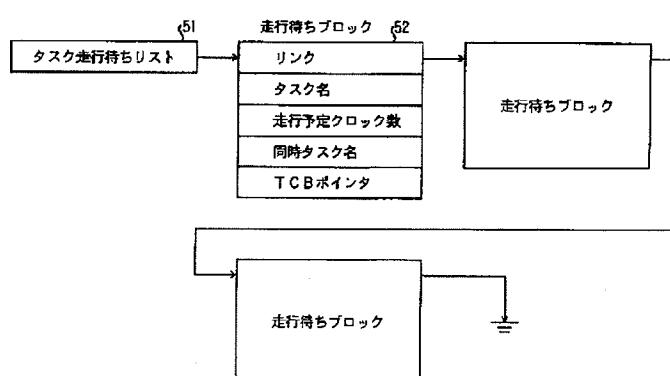
【図79】



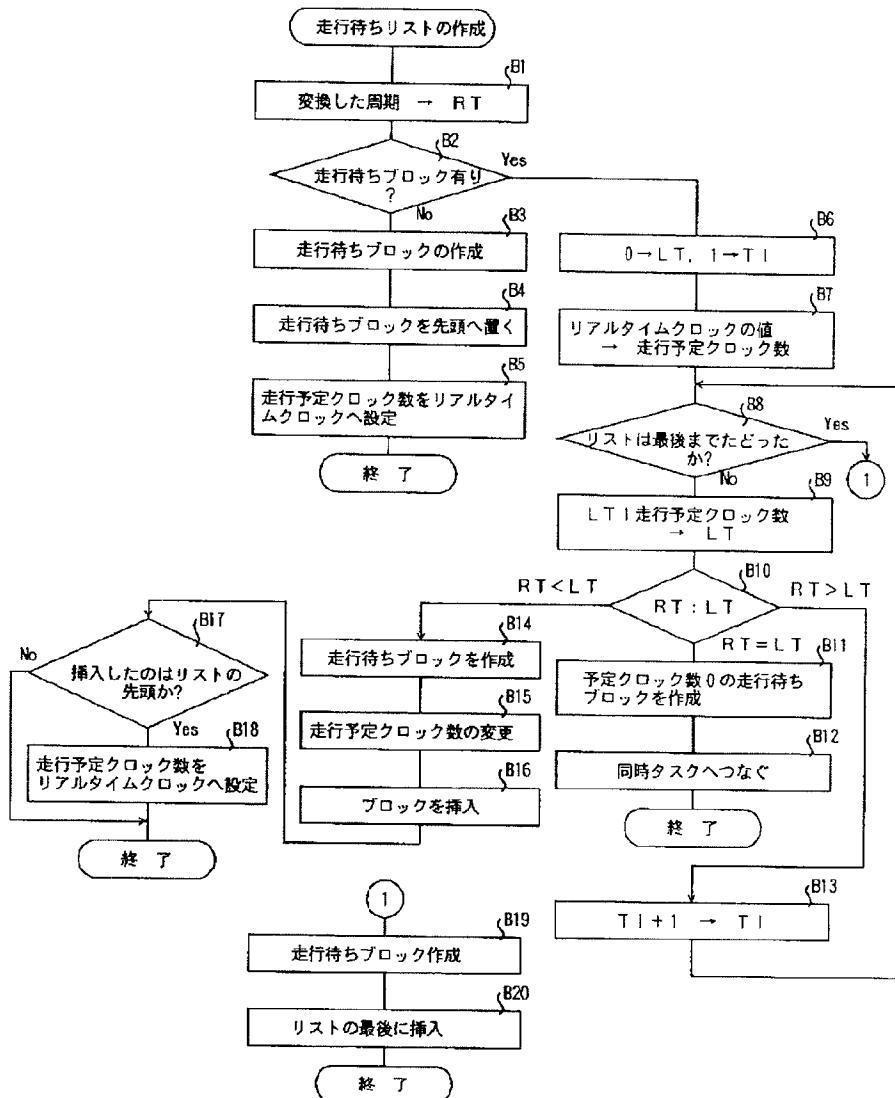
【図2】

タスク走行周期テーブル		
タスク名	走行周期	繰り返し回数
サーボ処理タスク	2.0	$\infty$
補正計算タスク	5.0	1
プログラム解析タスク	10.0	2
表示タスク	5.00	$\infty$
⋮	⋮	⋮

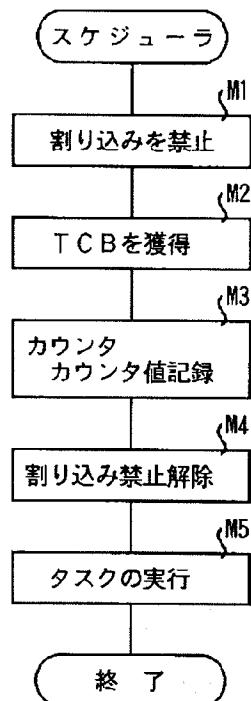
【図3】



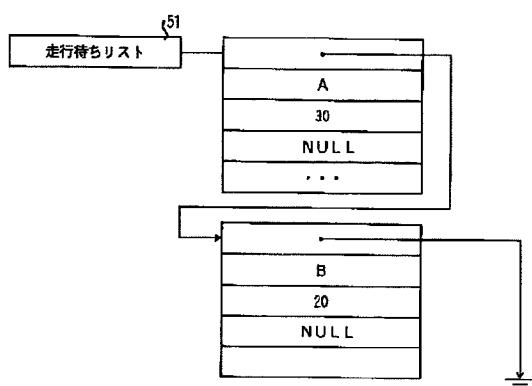
【図4】



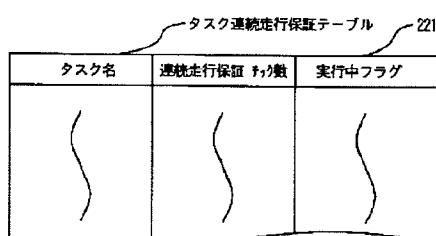
【図25】



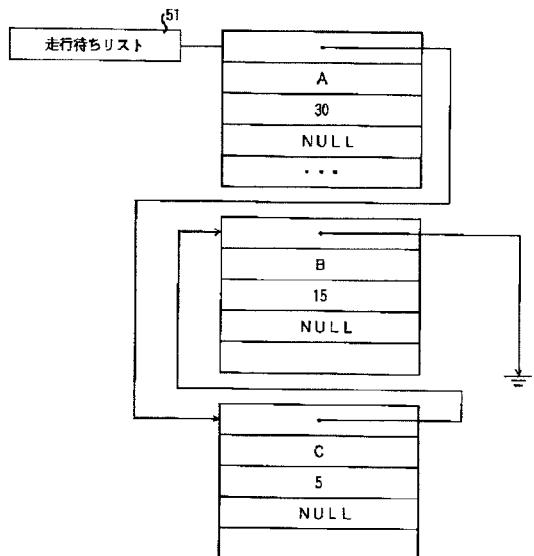
【図5】



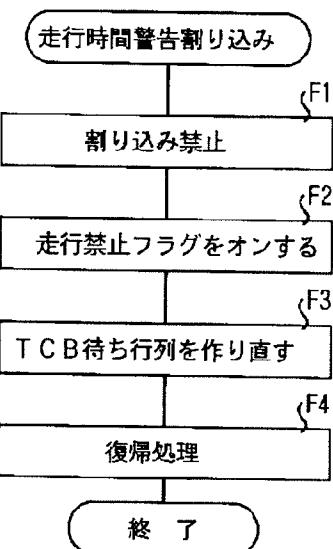
【図17】



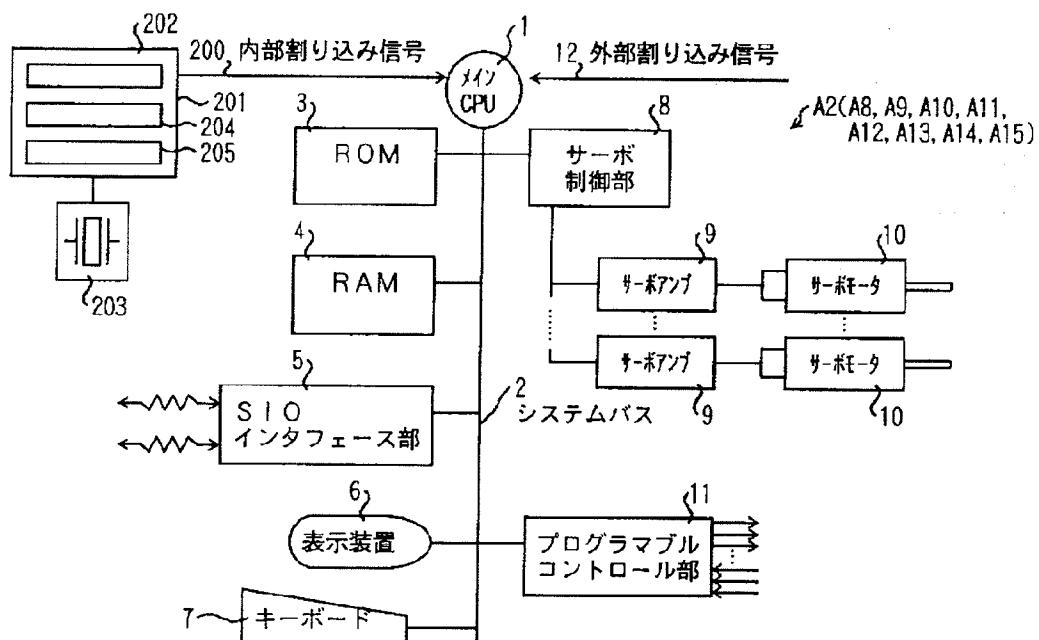
【図6】



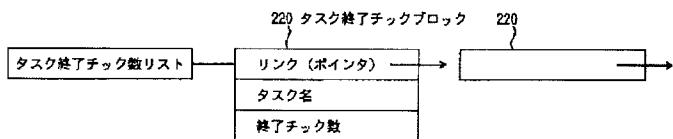
【图12】



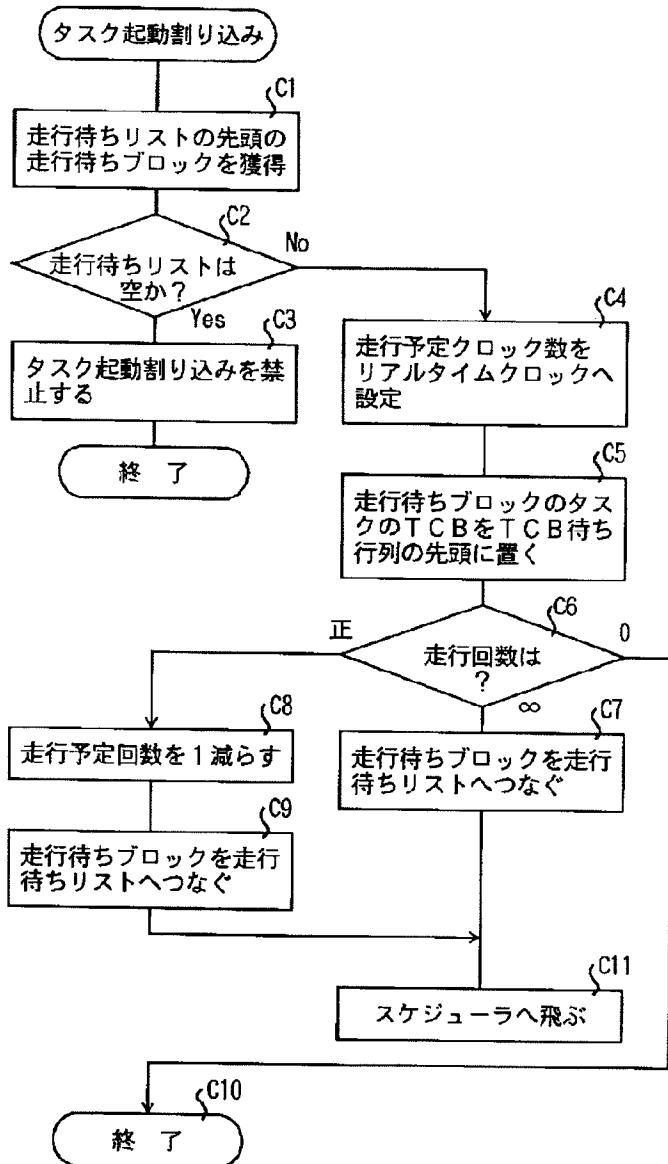
[图9]



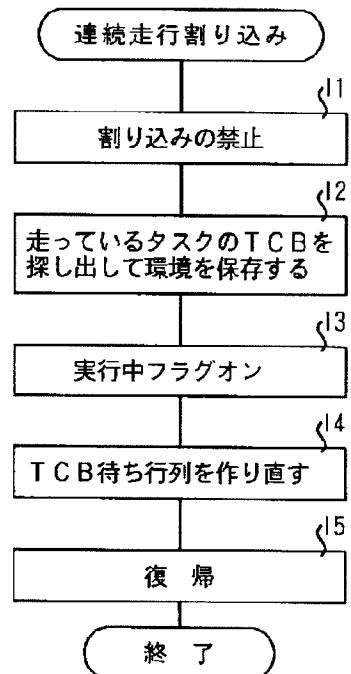
### 【図16】



【図7】



【図18】



【図31】

250 タスク走行標準データ		
タスク名	走行割合	全走行チック数

251  
走行時間判定許容度

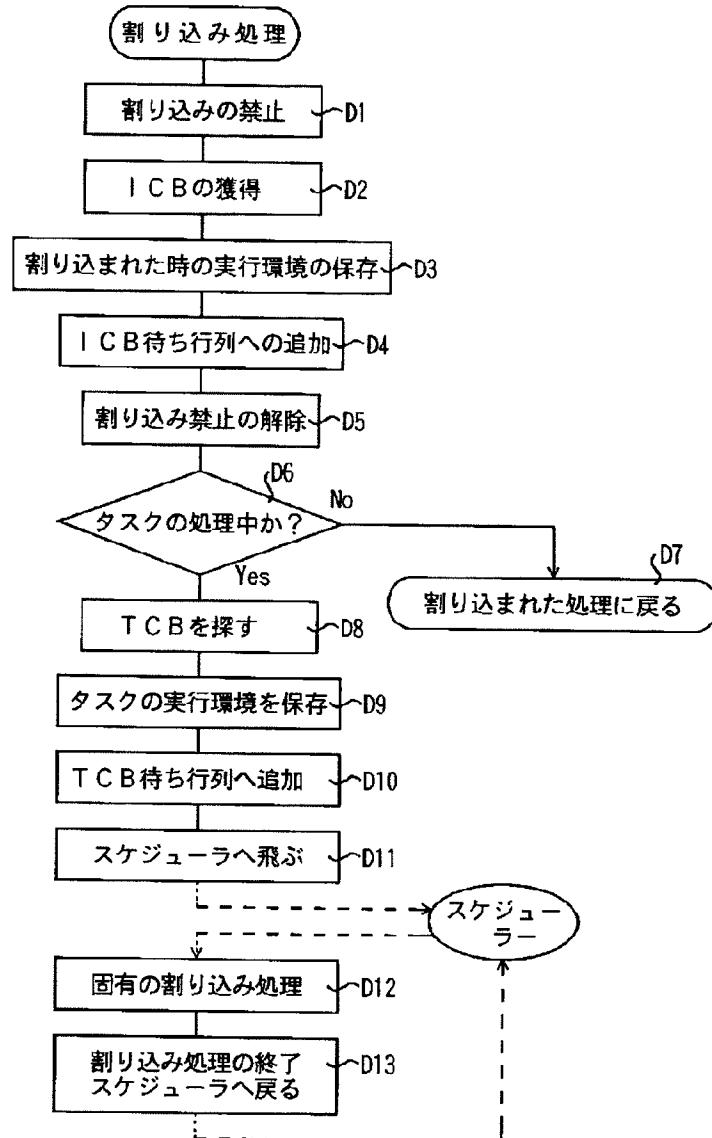
上限	下限

【図22】

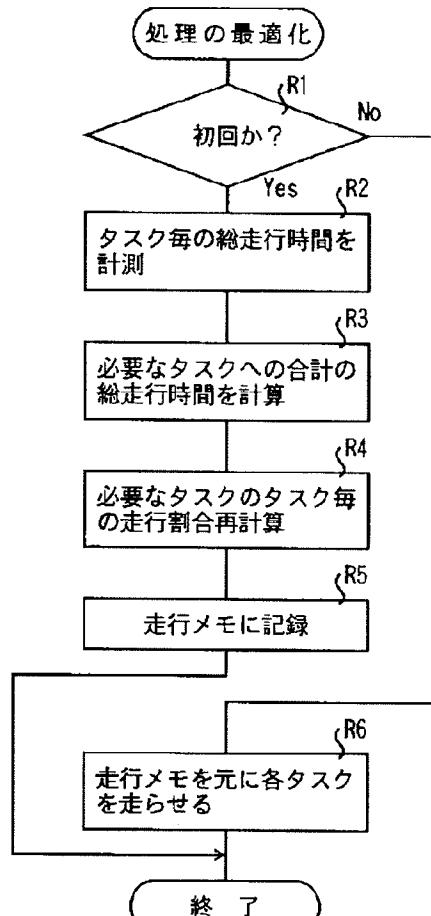
230 カウンタ値メモ	
タスク名	合計チック数

231 タスク走行時間テーブル

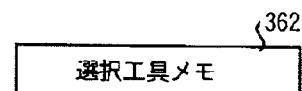
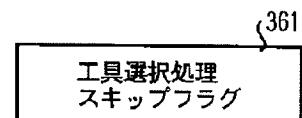
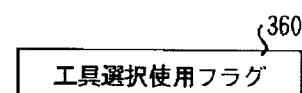
【図8】



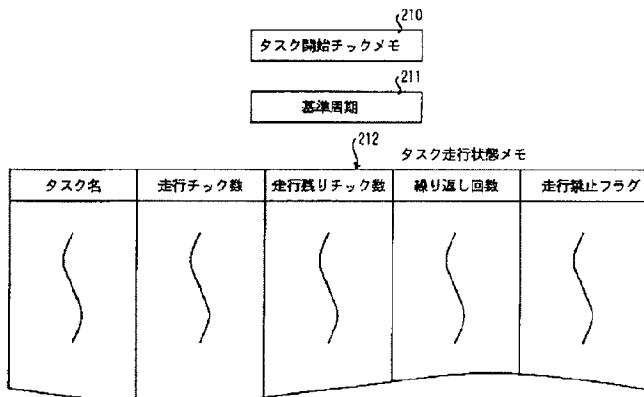
【図30】



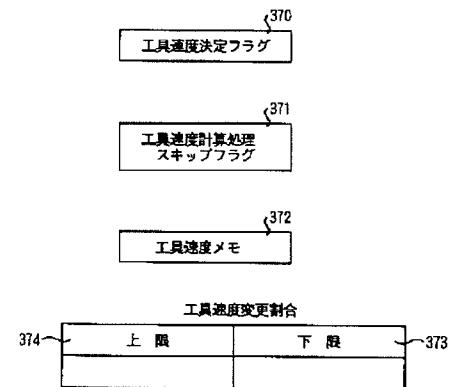
【図33】



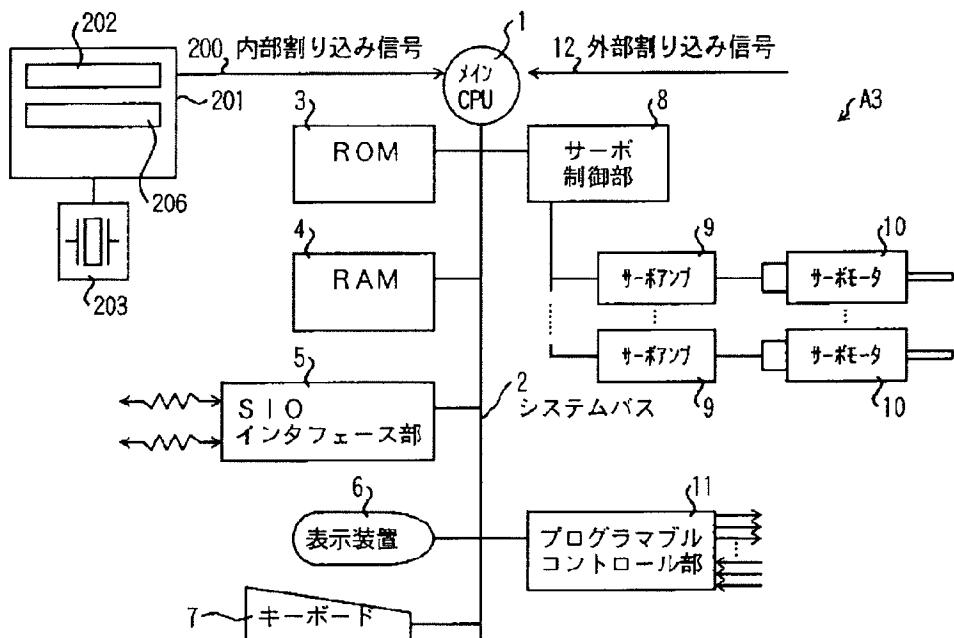
【図10】



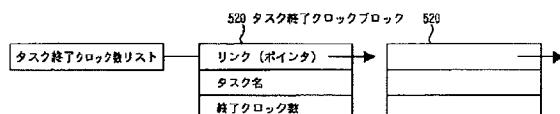
【図35】



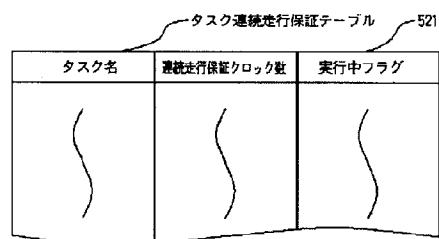
【図15】



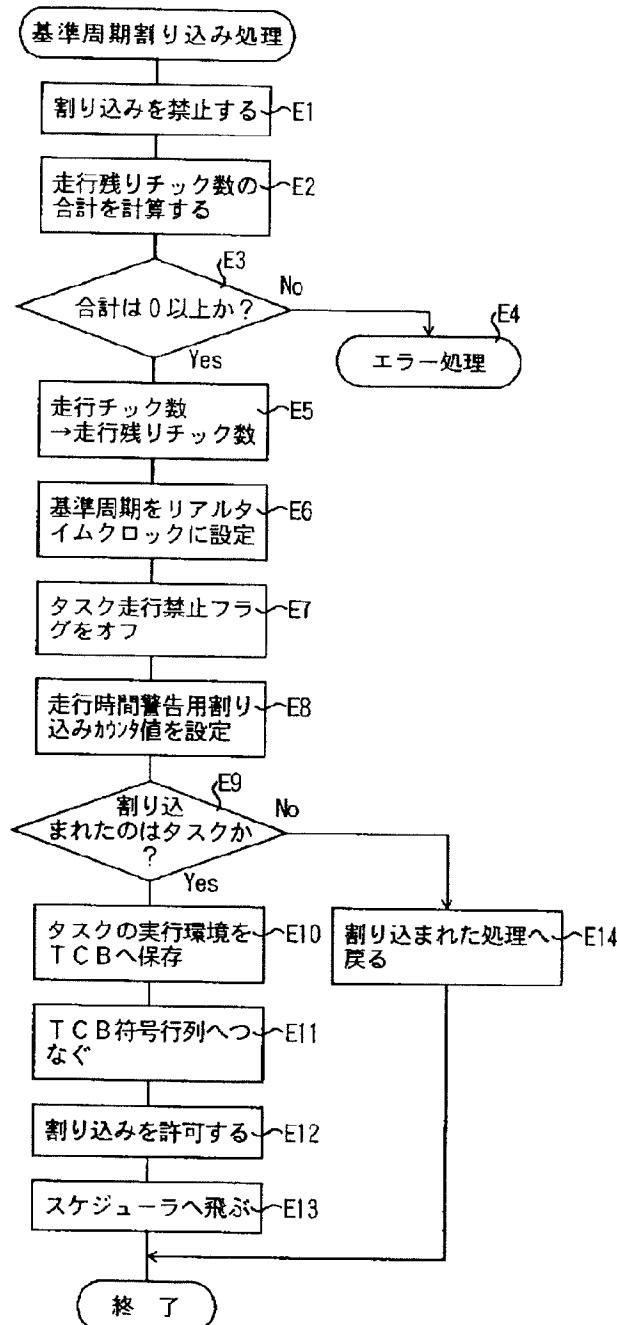
【図55】



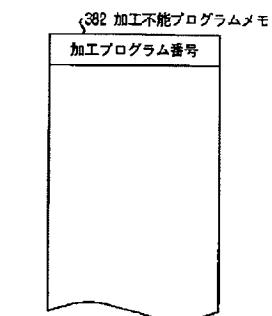
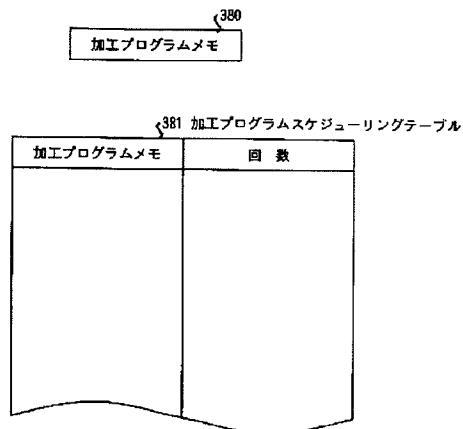
【図56】



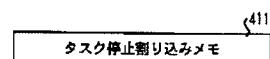
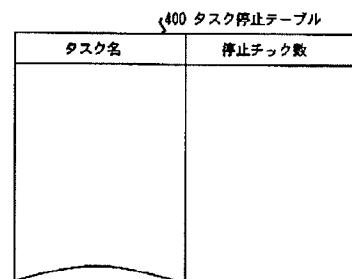
【図11】



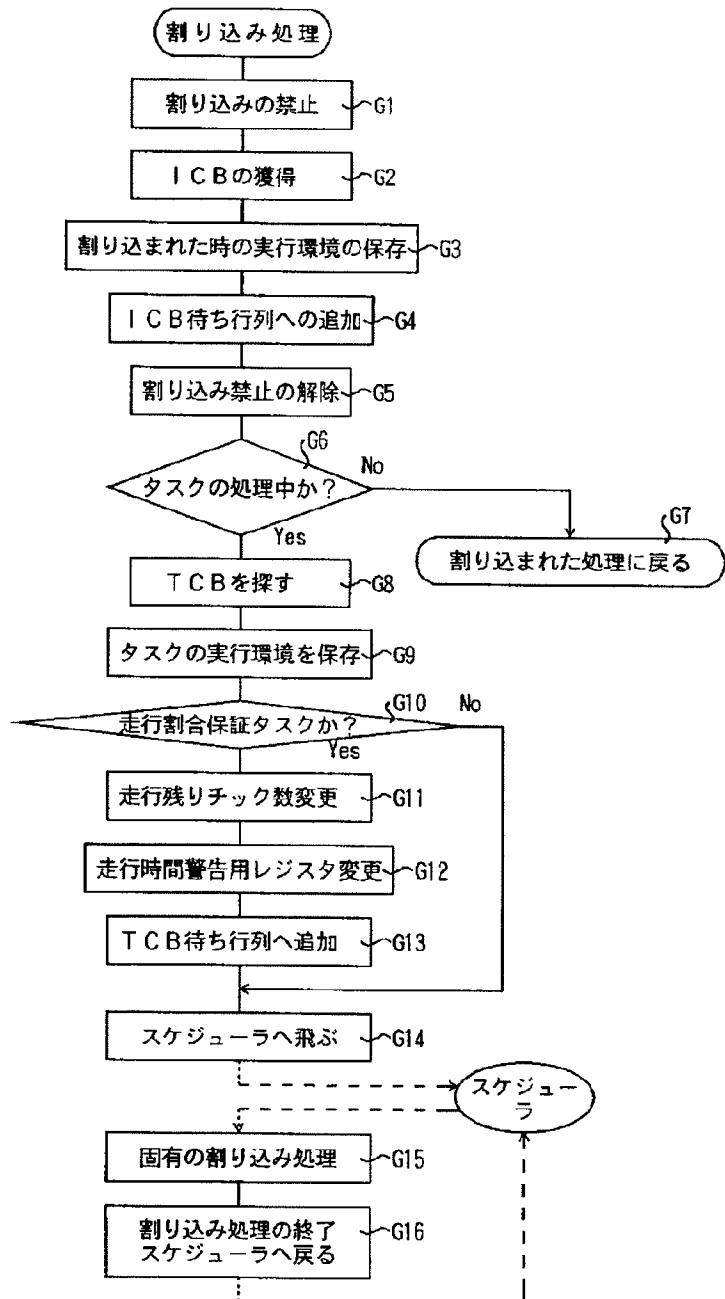
【図37】



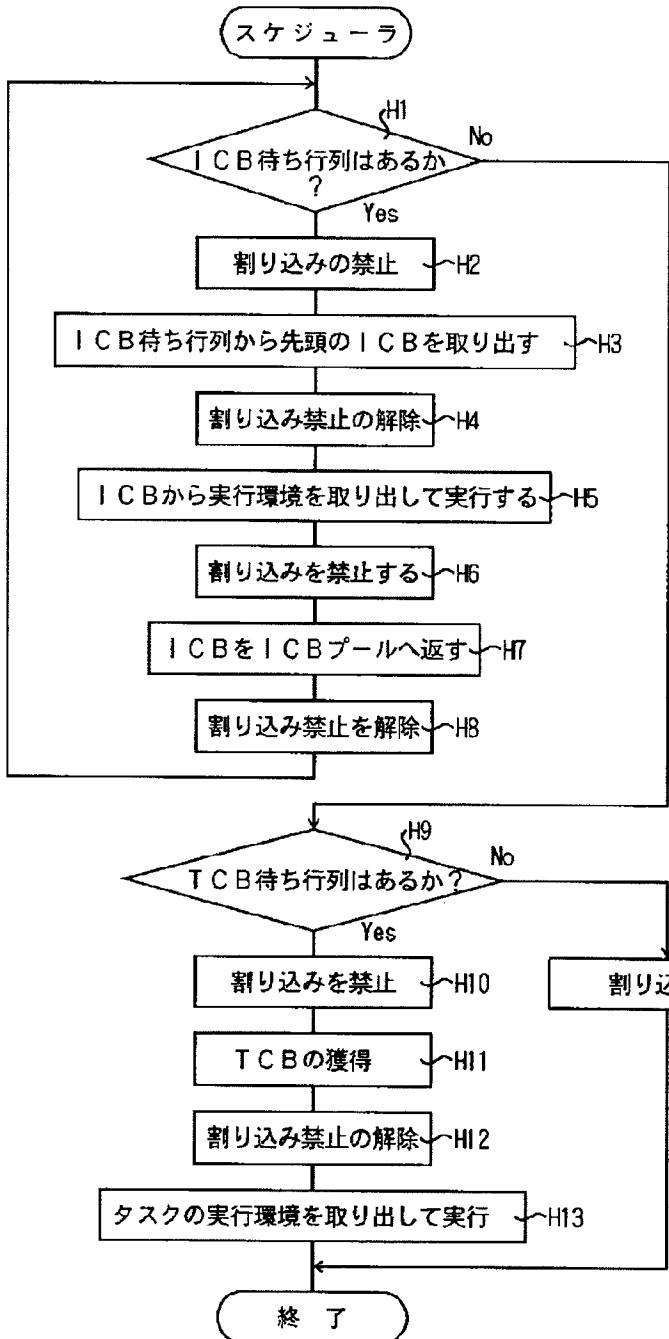
【図39】



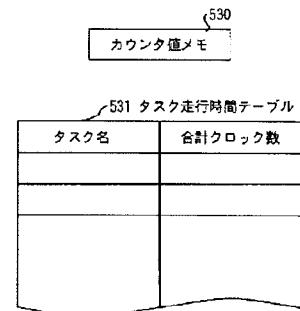
【図13】



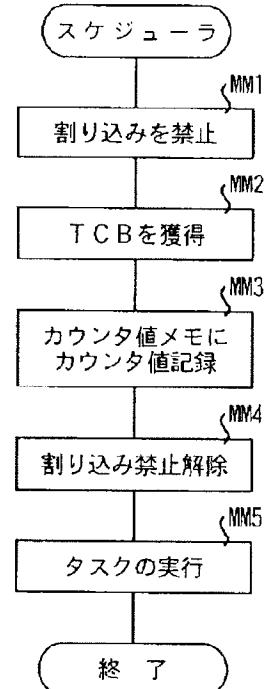
【図14】



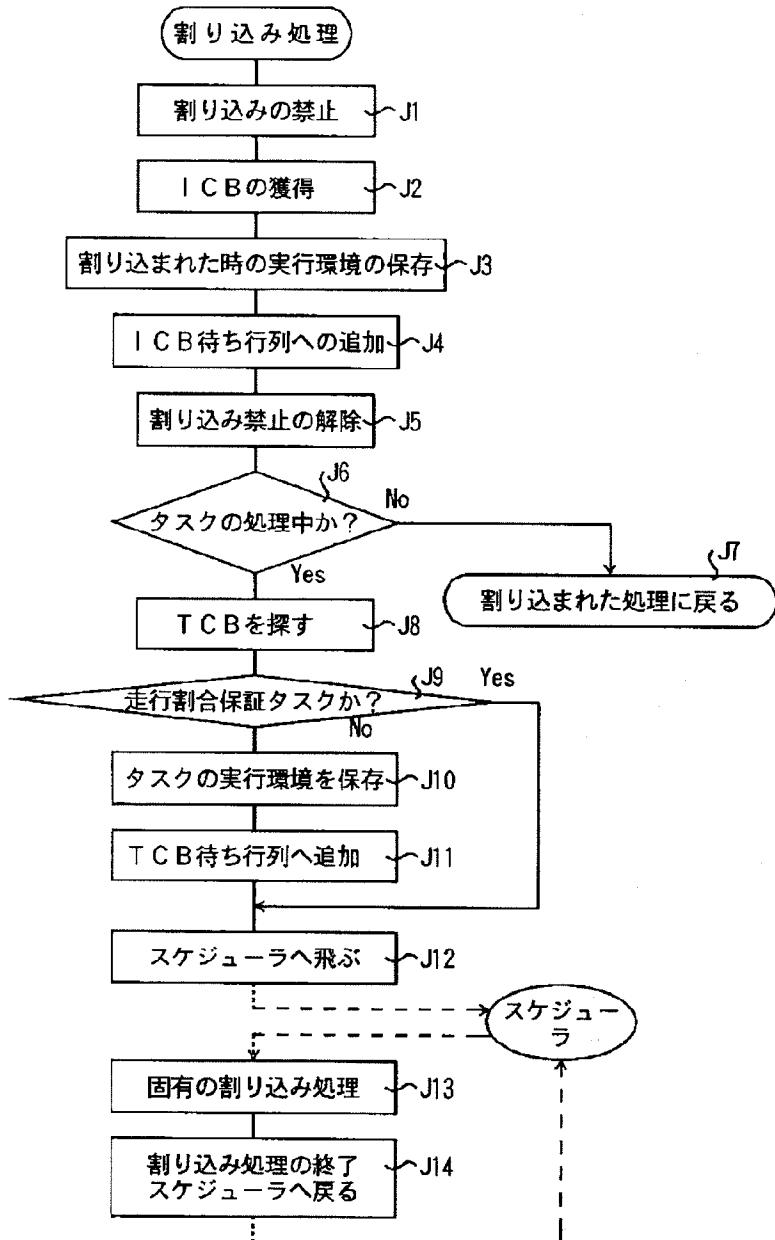
【図61】



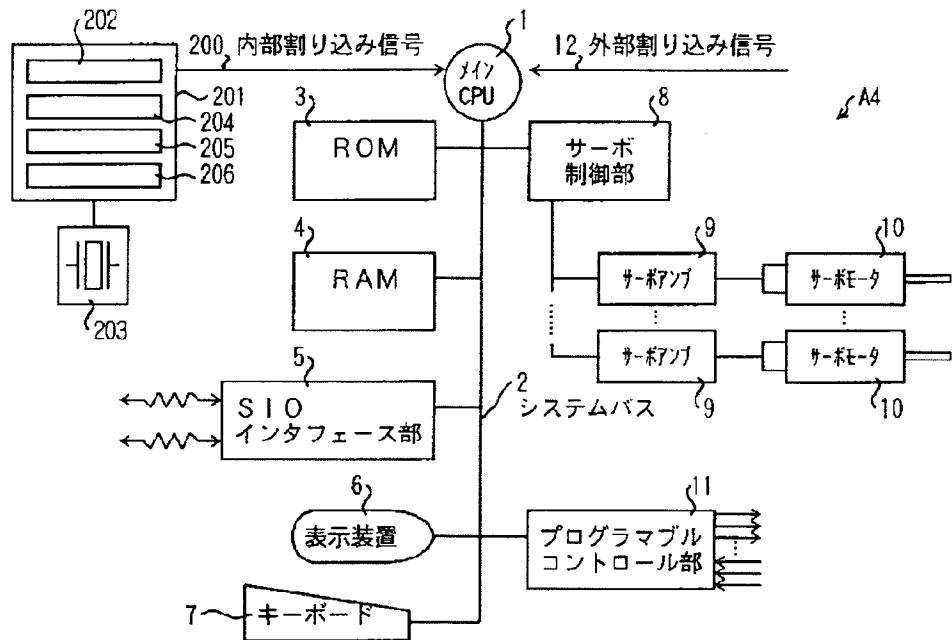
【図64】



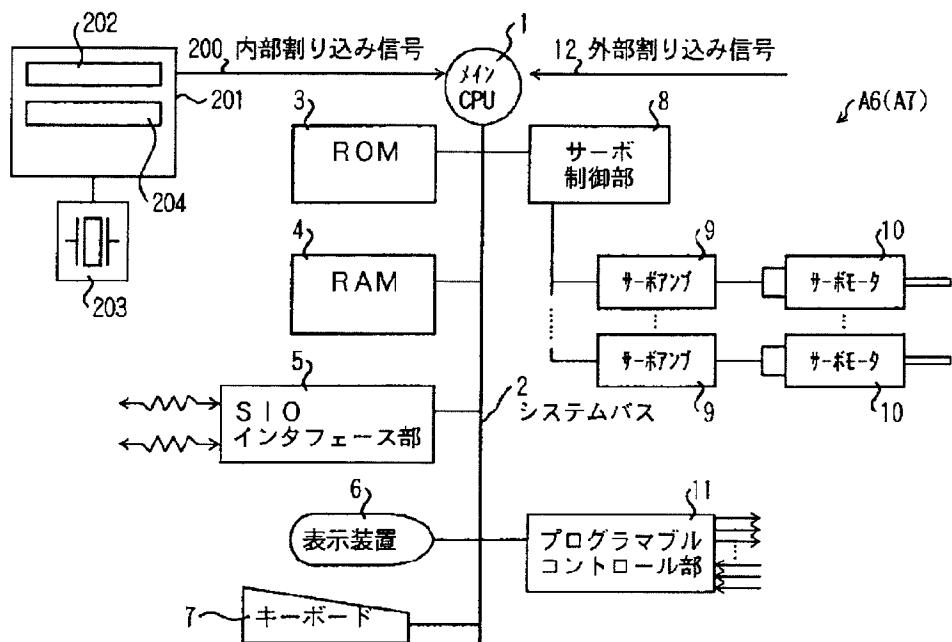
【図19】



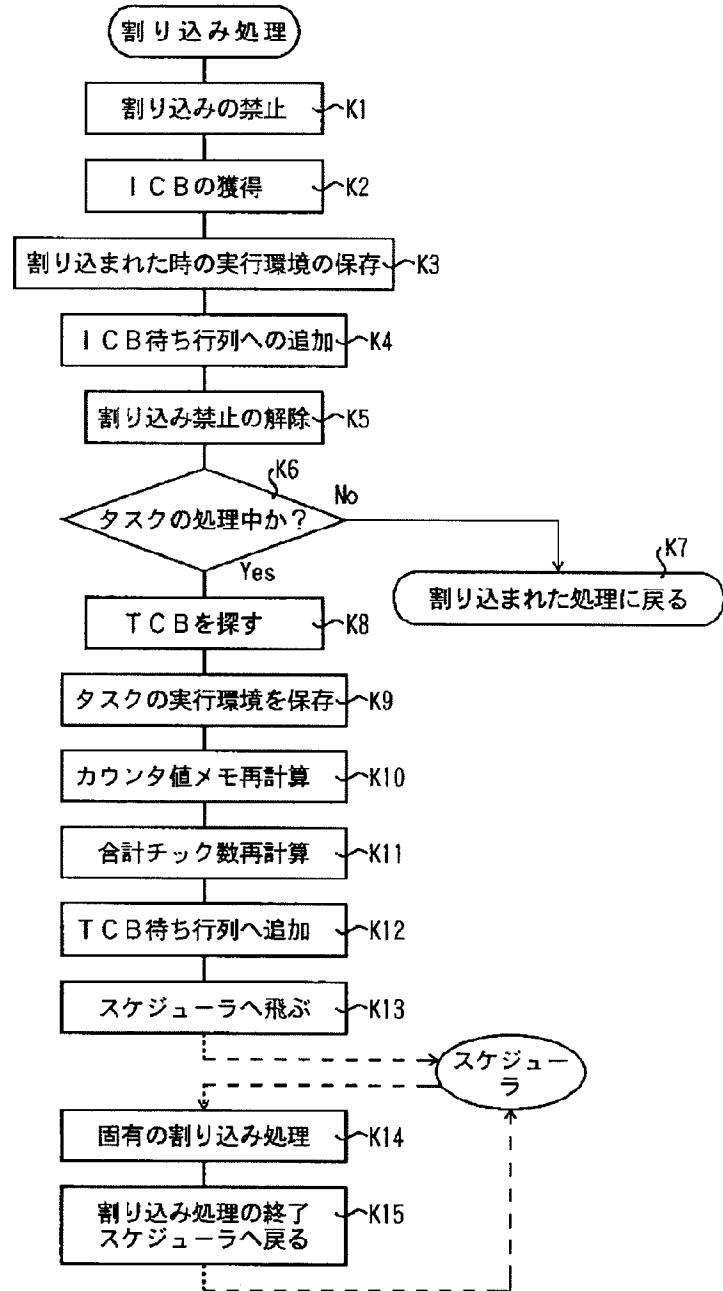
【図20】



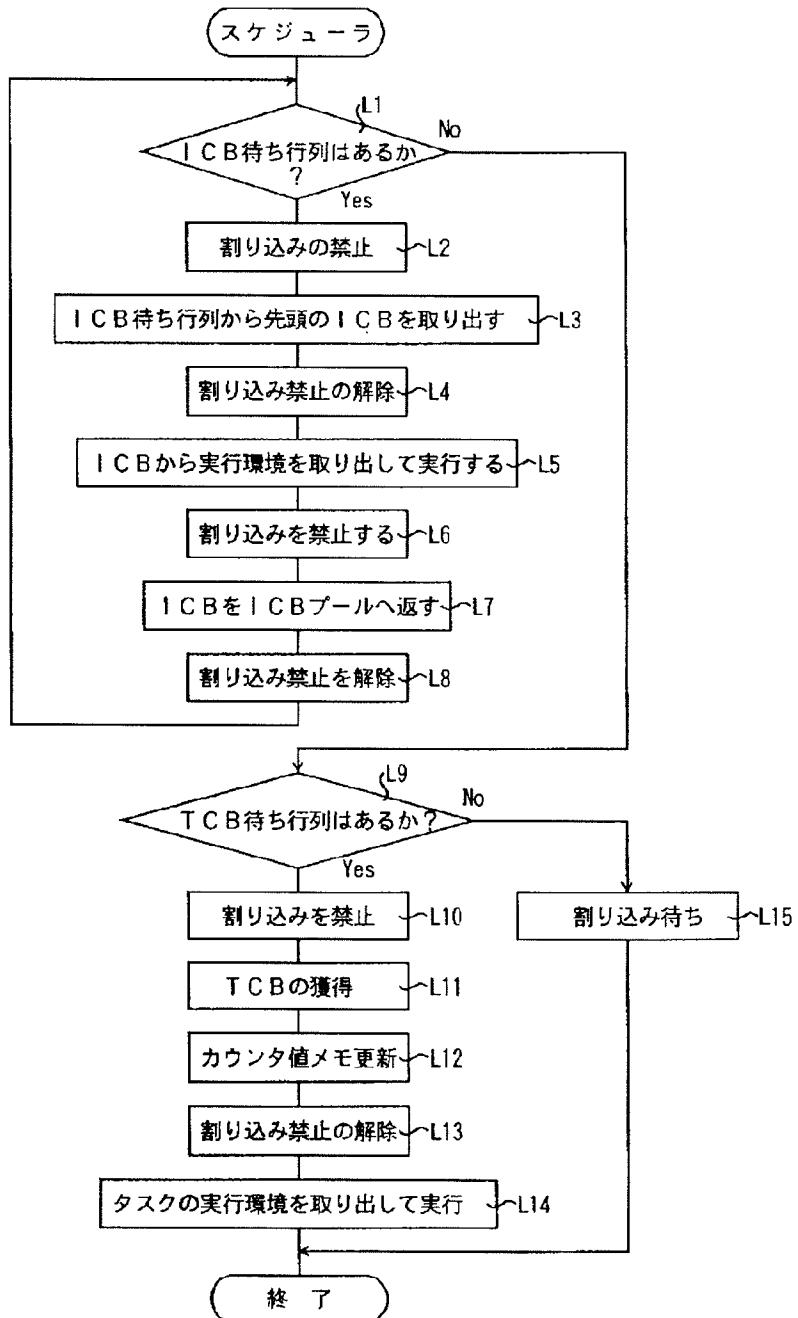
【図21】



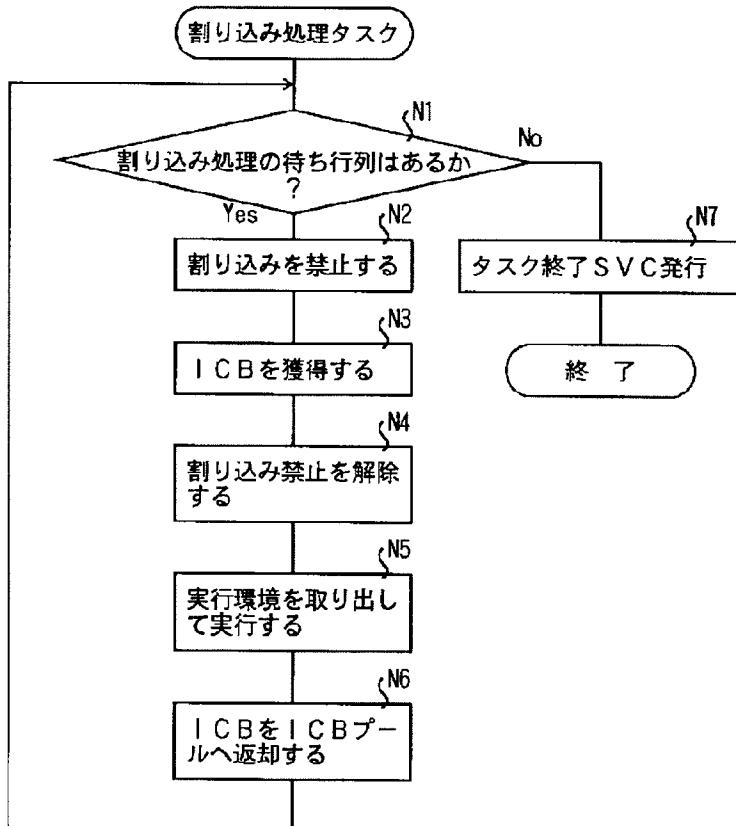
【図23】



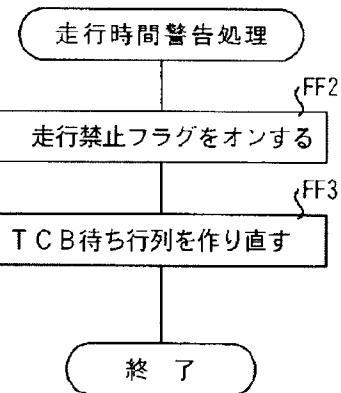
【図24】



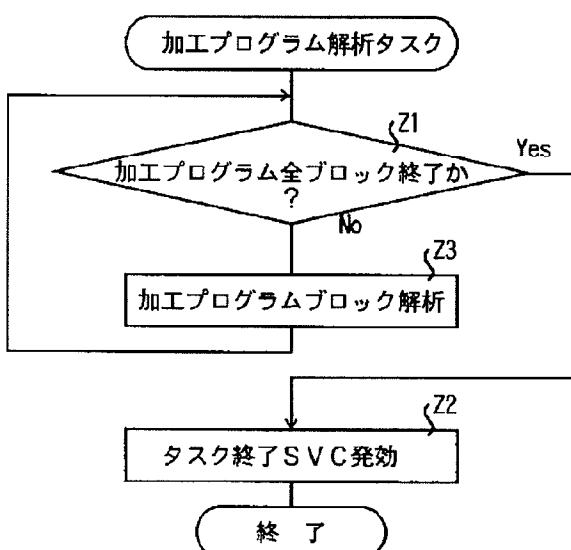
【図26】



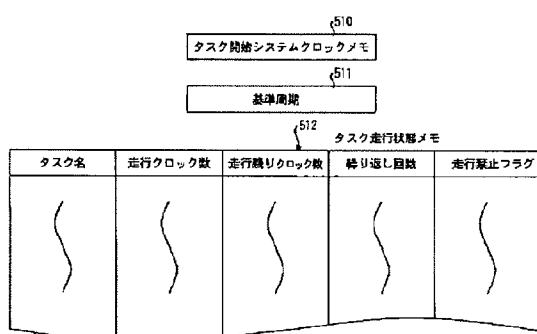
【図51】



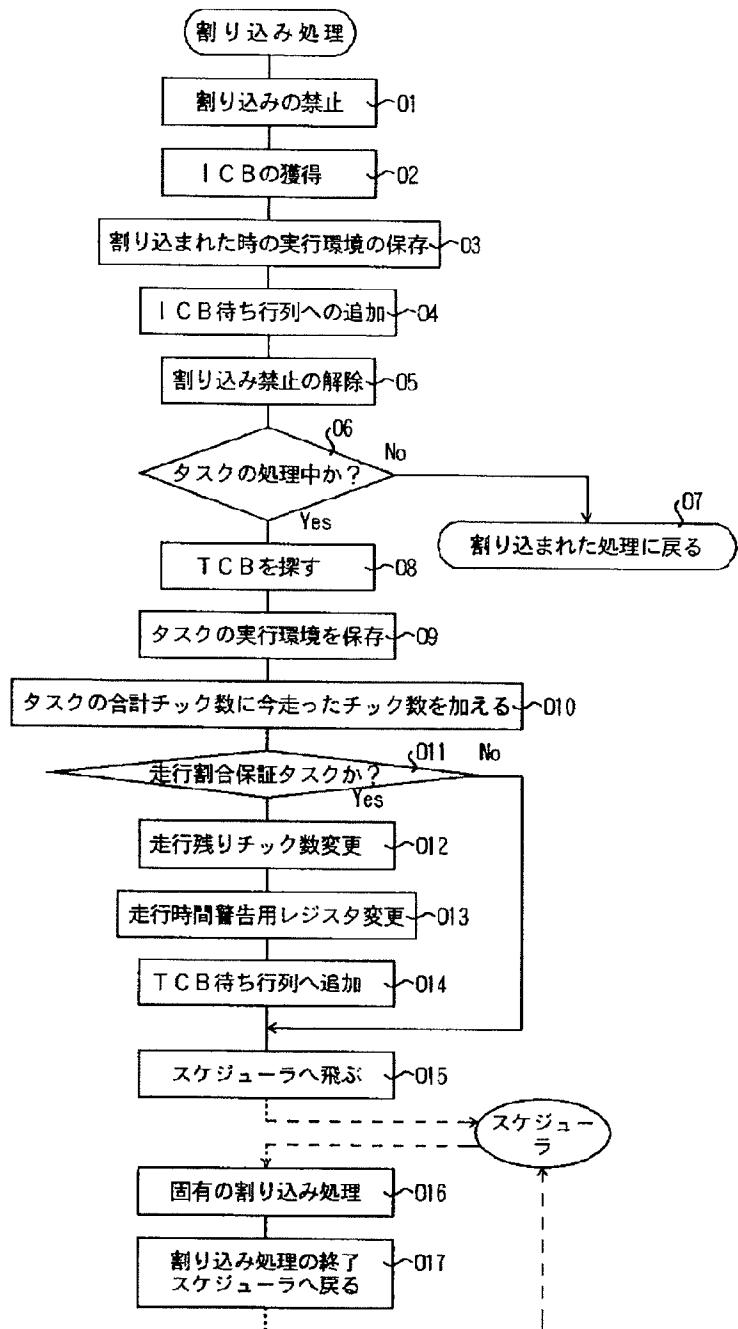
【図43】



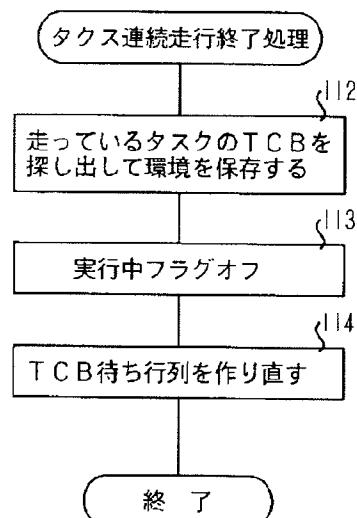
【図49】



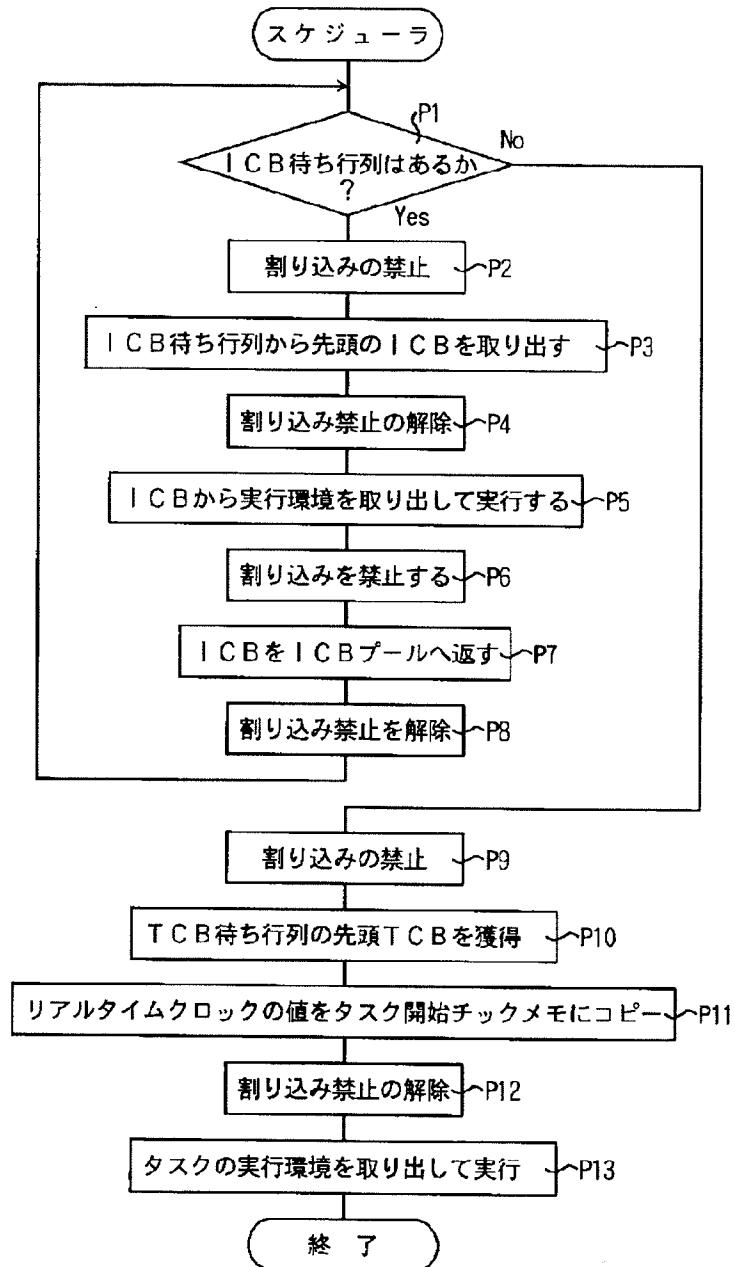
【図27】



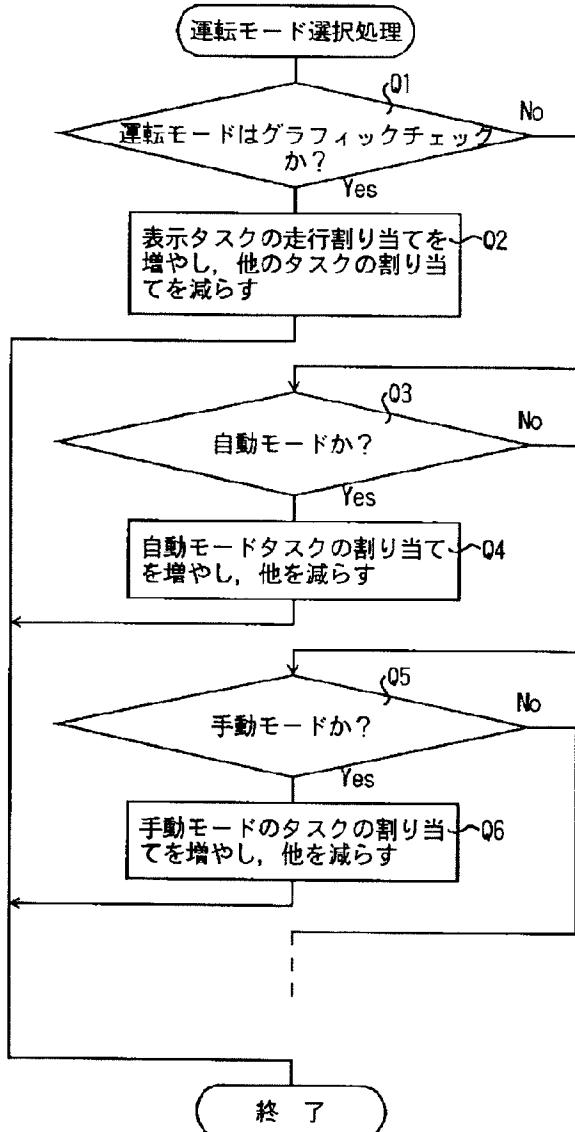
【図57】



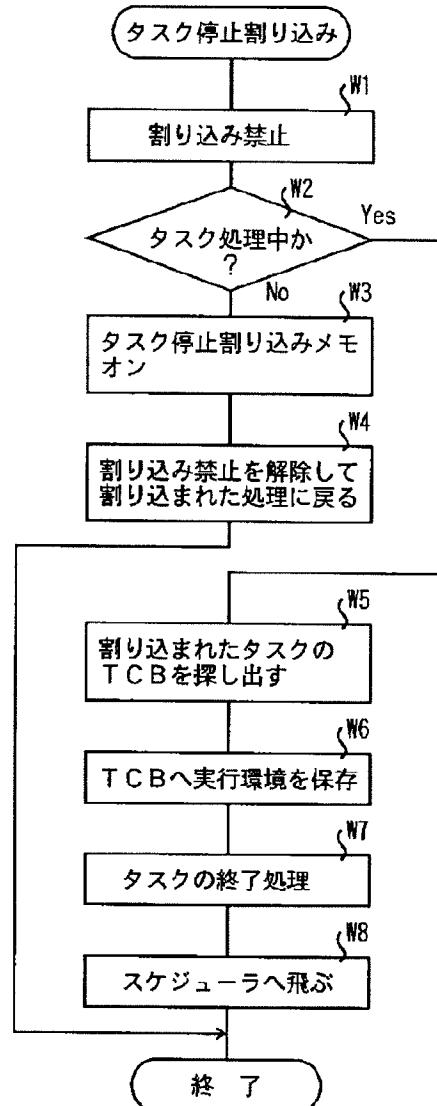
【図28】



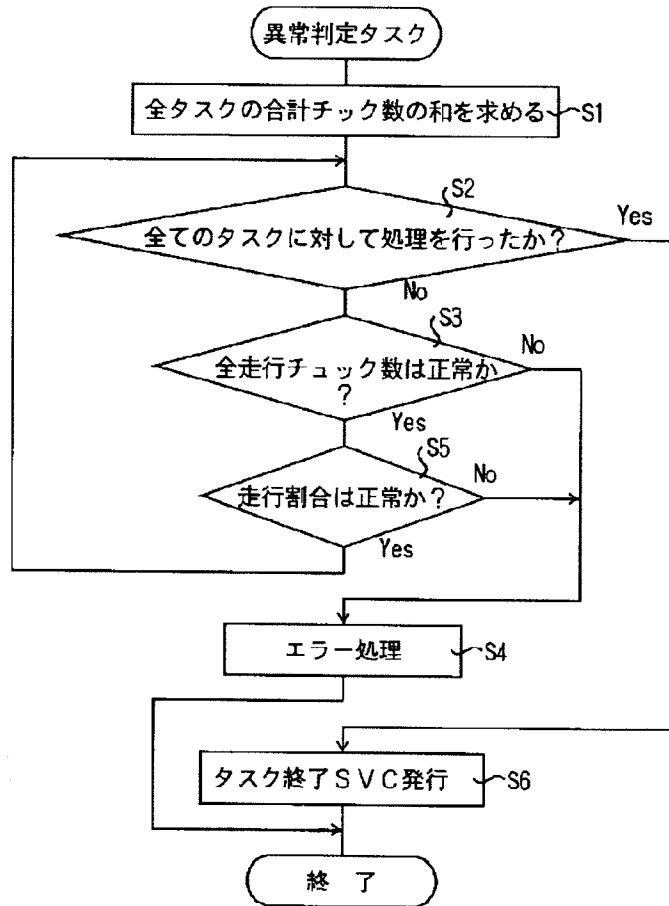
【図29】



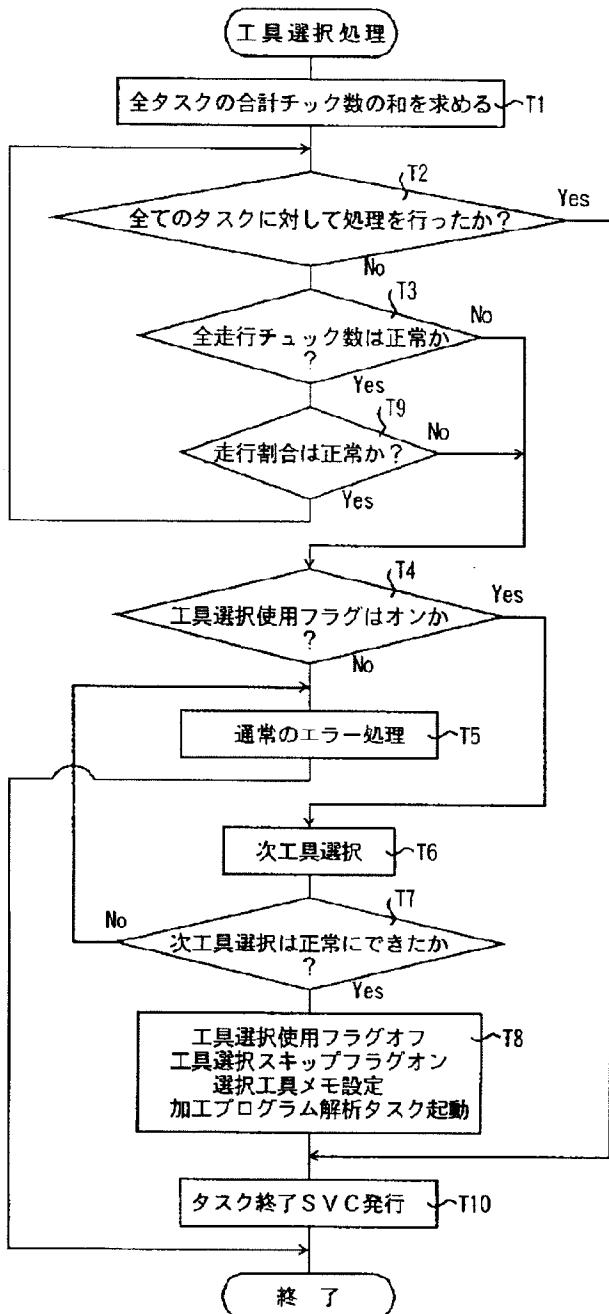
【図40】



【図32】



【図34】



【図67】

550 タスク走行標準データ

タスク名	走行割合	全走行チック数

551 走行時間判定許容度

552 走行割合判定許容度

上限	下限

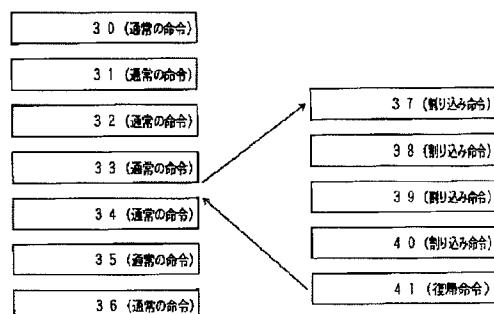
【図69】

600 タスク停止テーブル

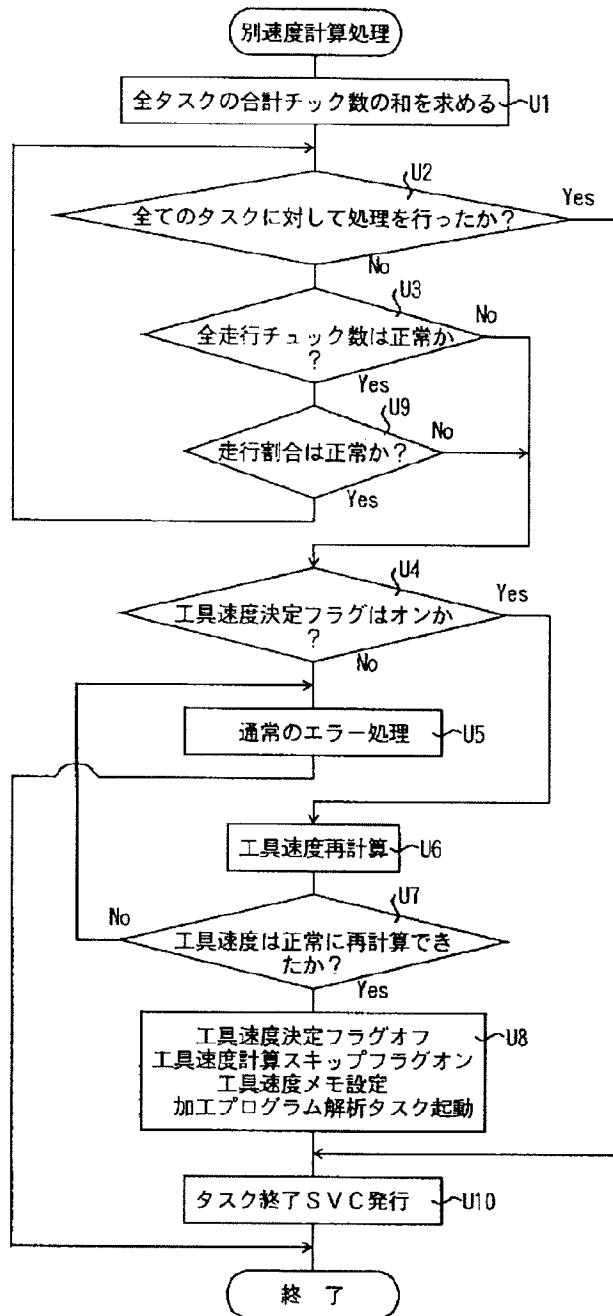
タスク名	停止クロック数

611 タスク停止クロックメモ

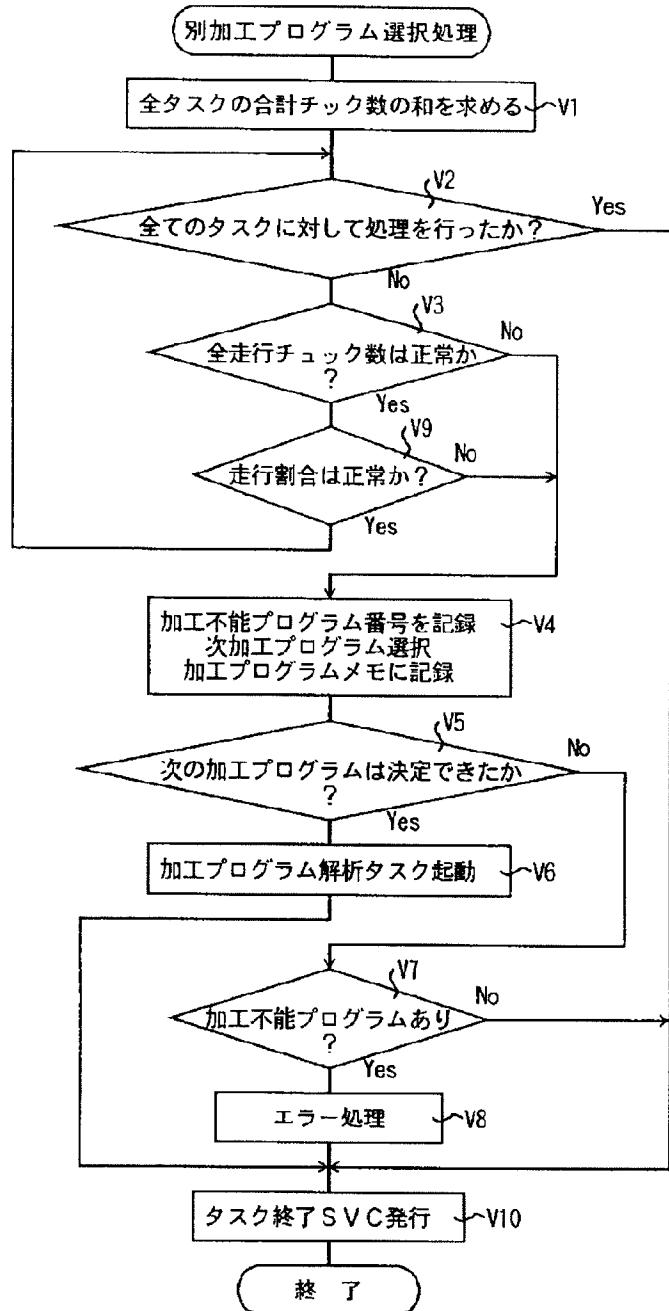
【図75】



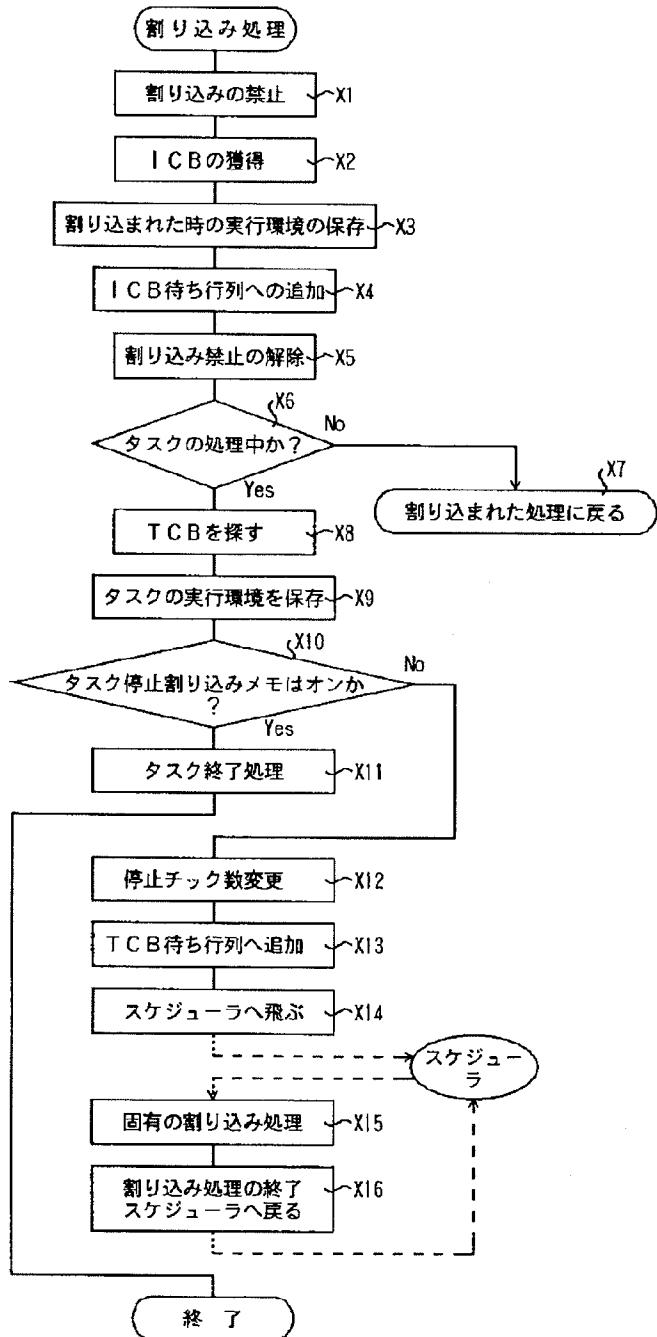
【図36】



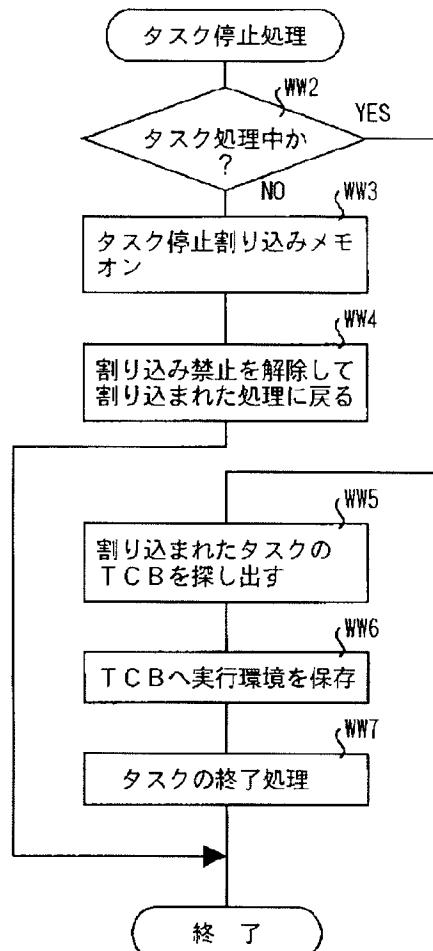
【図38】



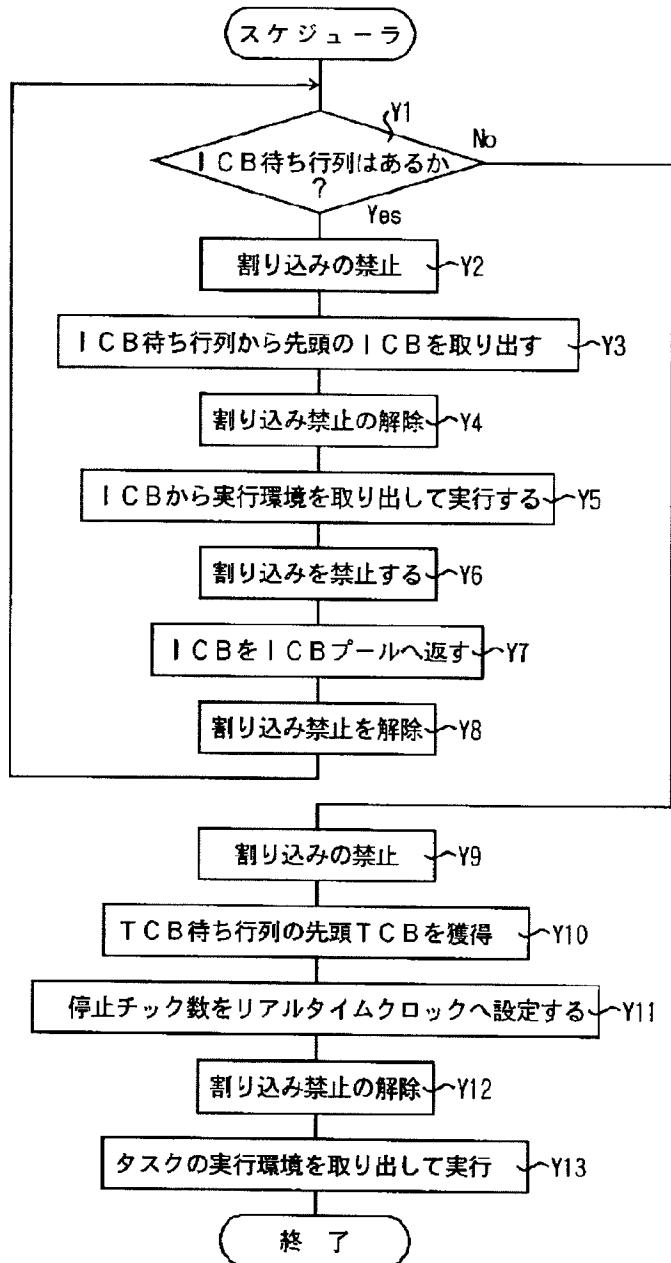
【図41】



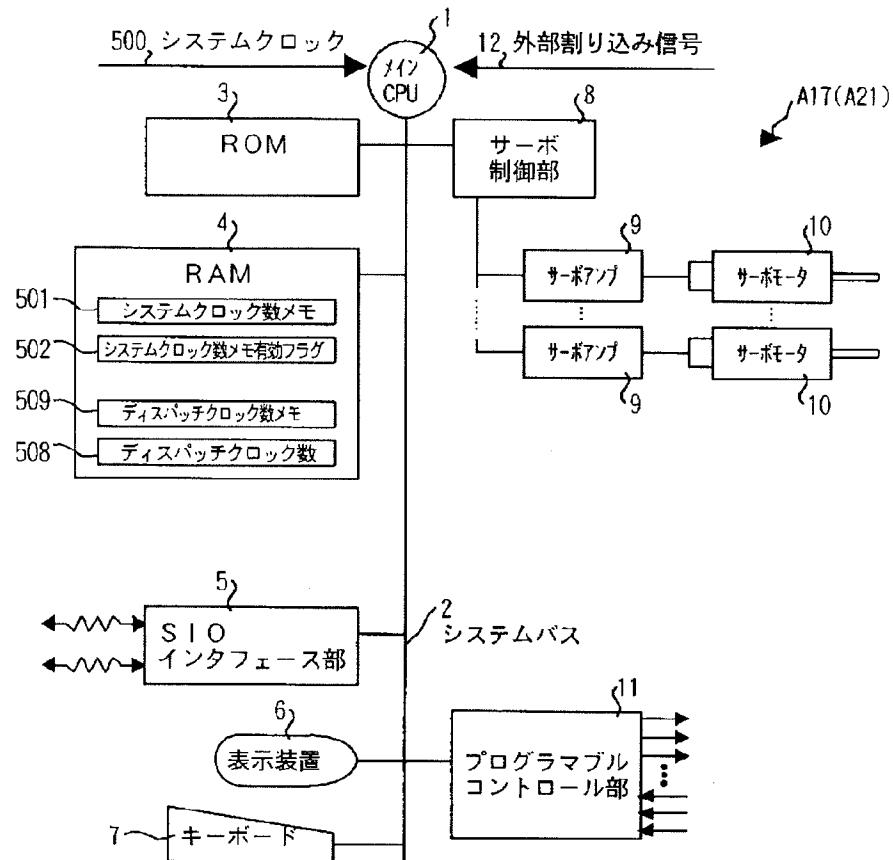
【図70】



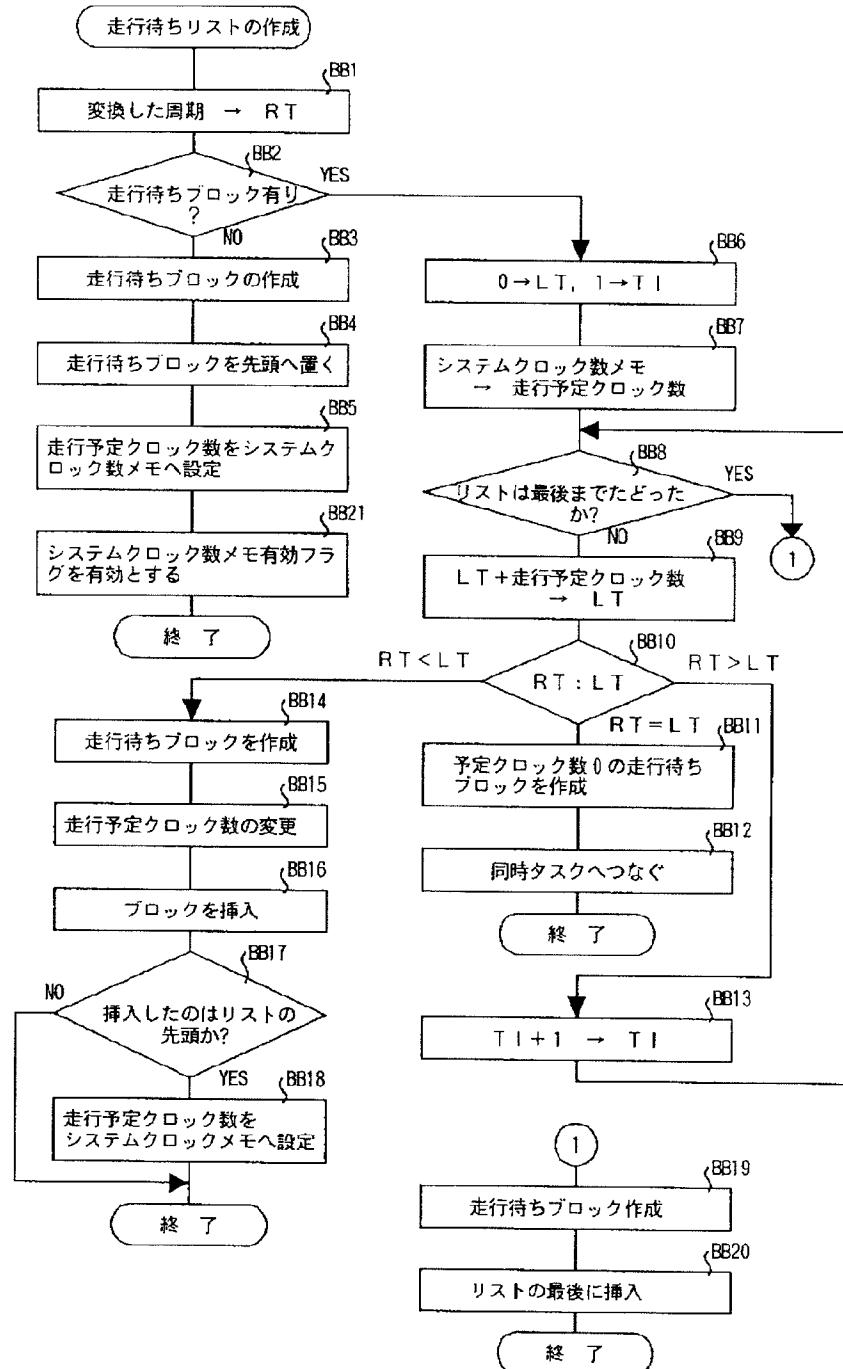
【図42】



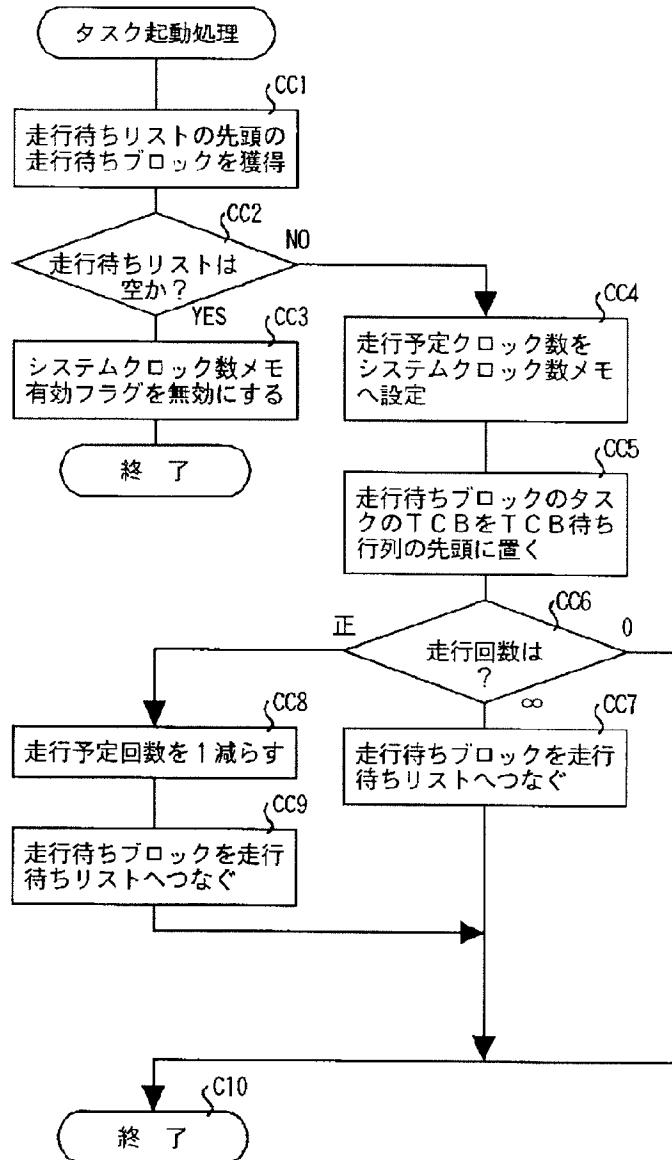
【図44】



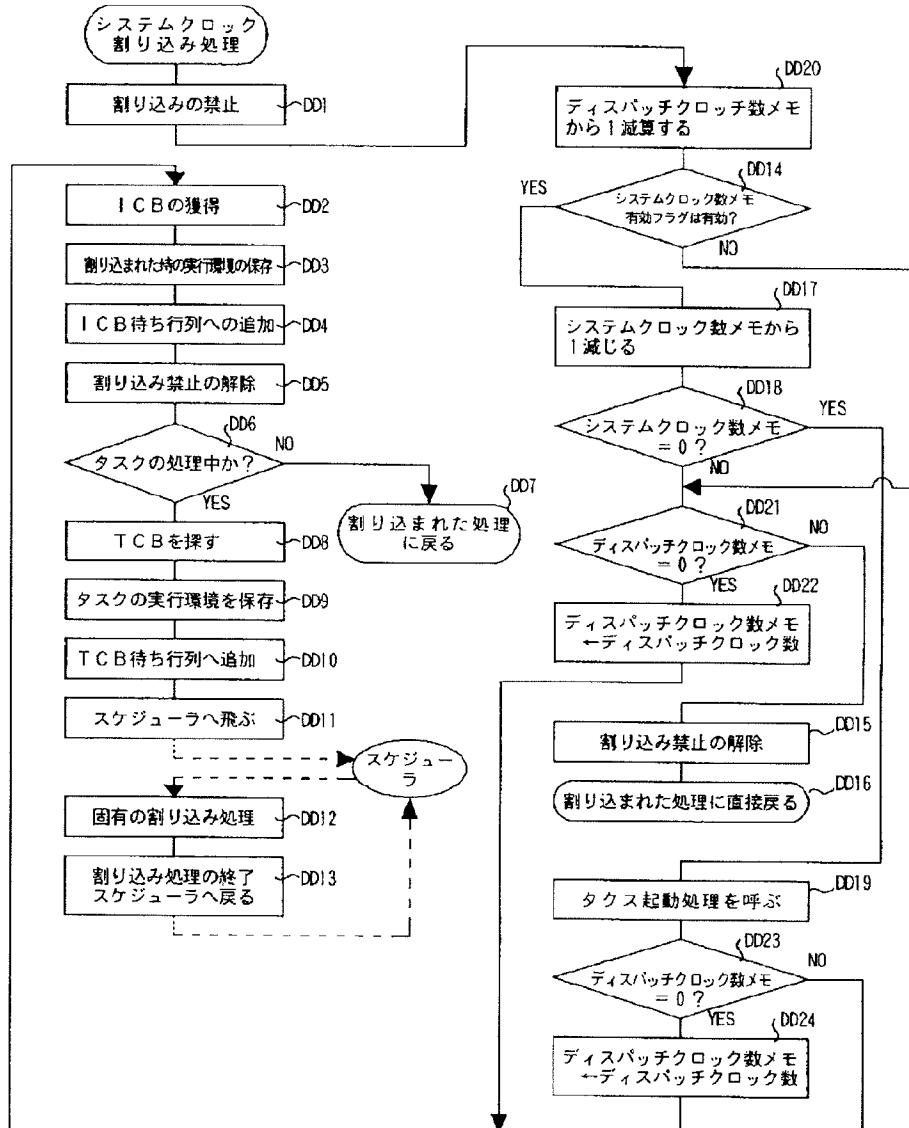
【図45】



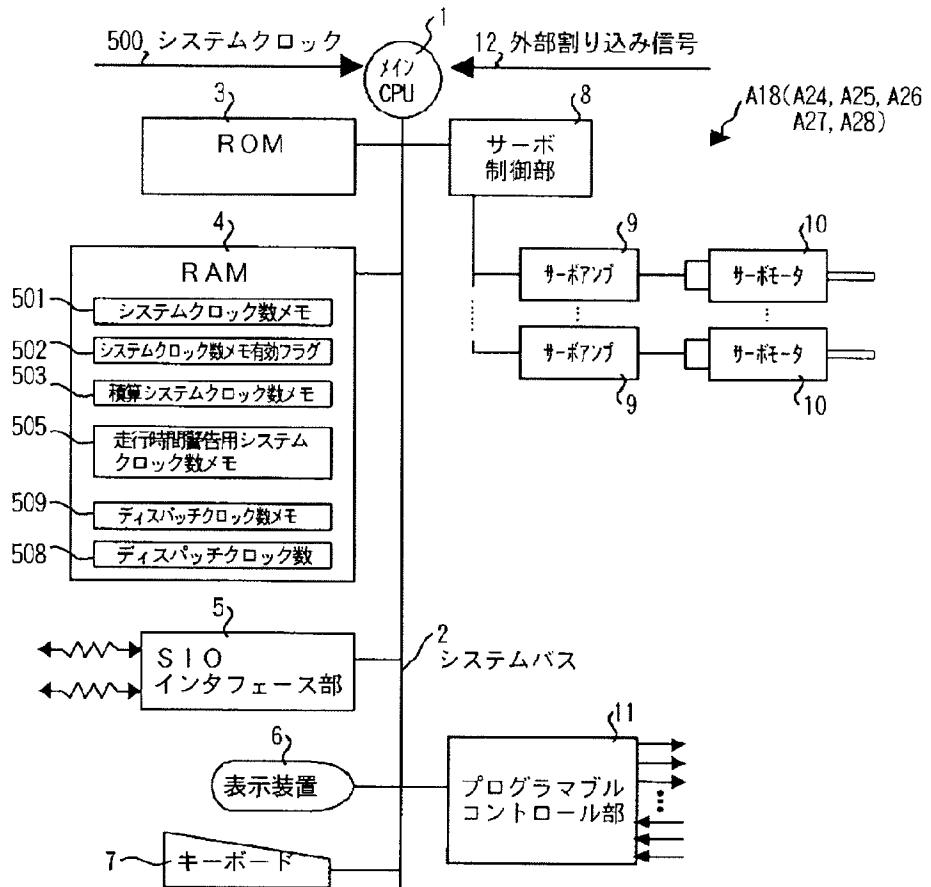
【図46】



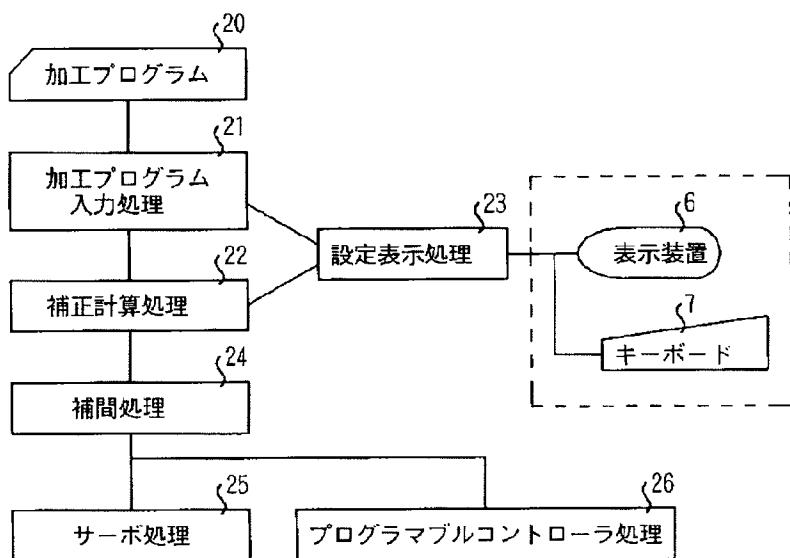
【図47】



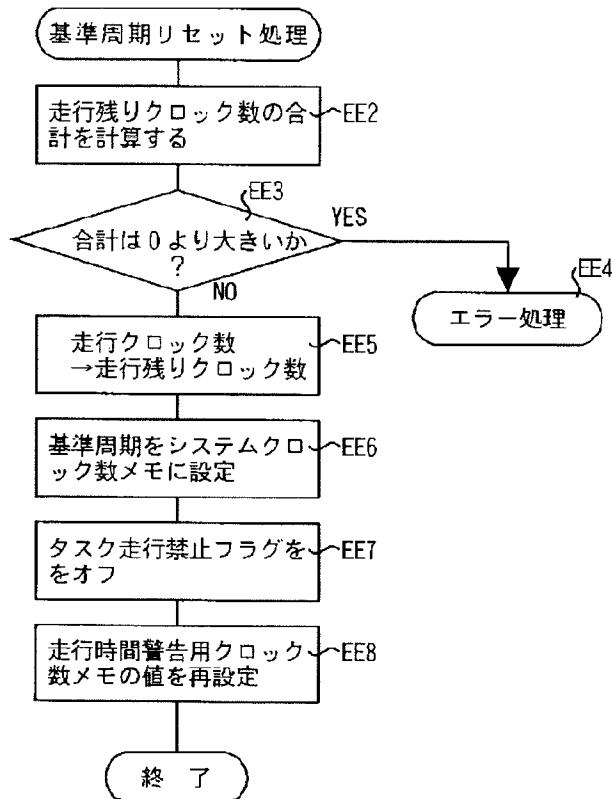
【図48】



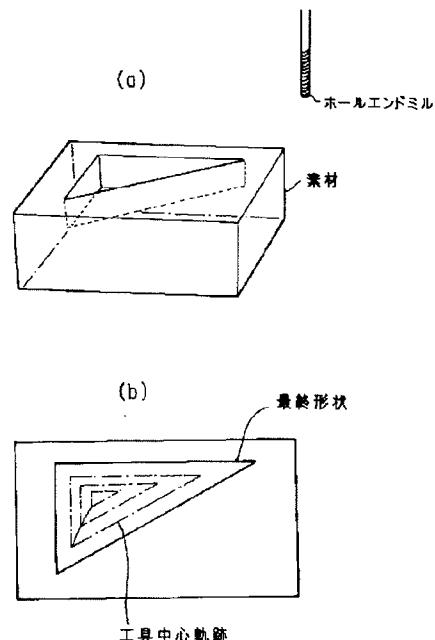
【図74】



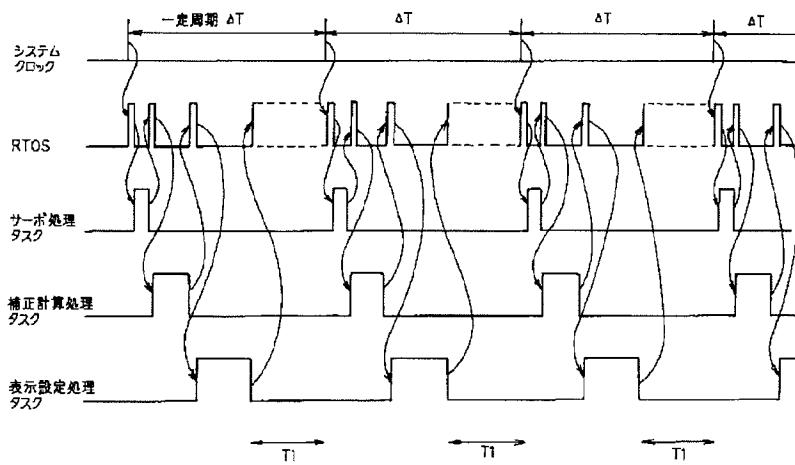
【図50】



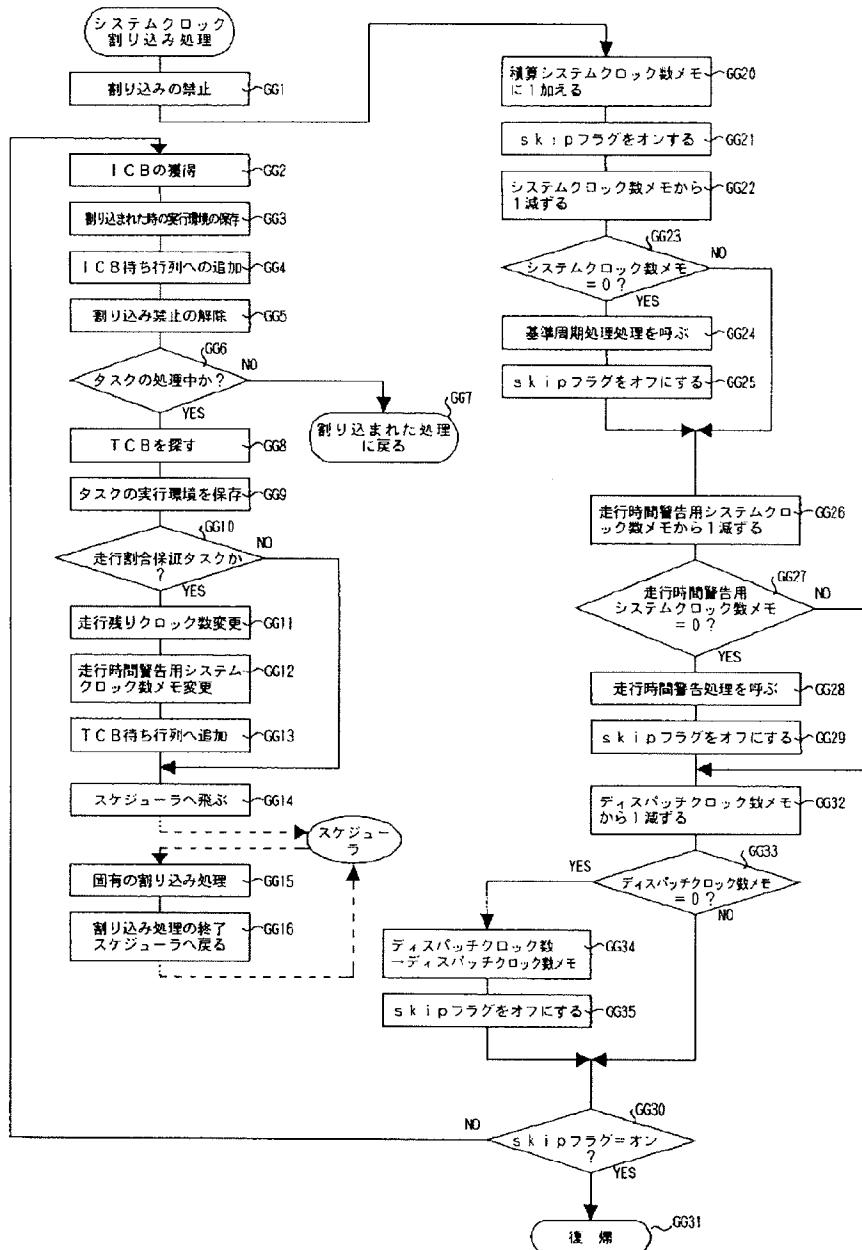
【図78】



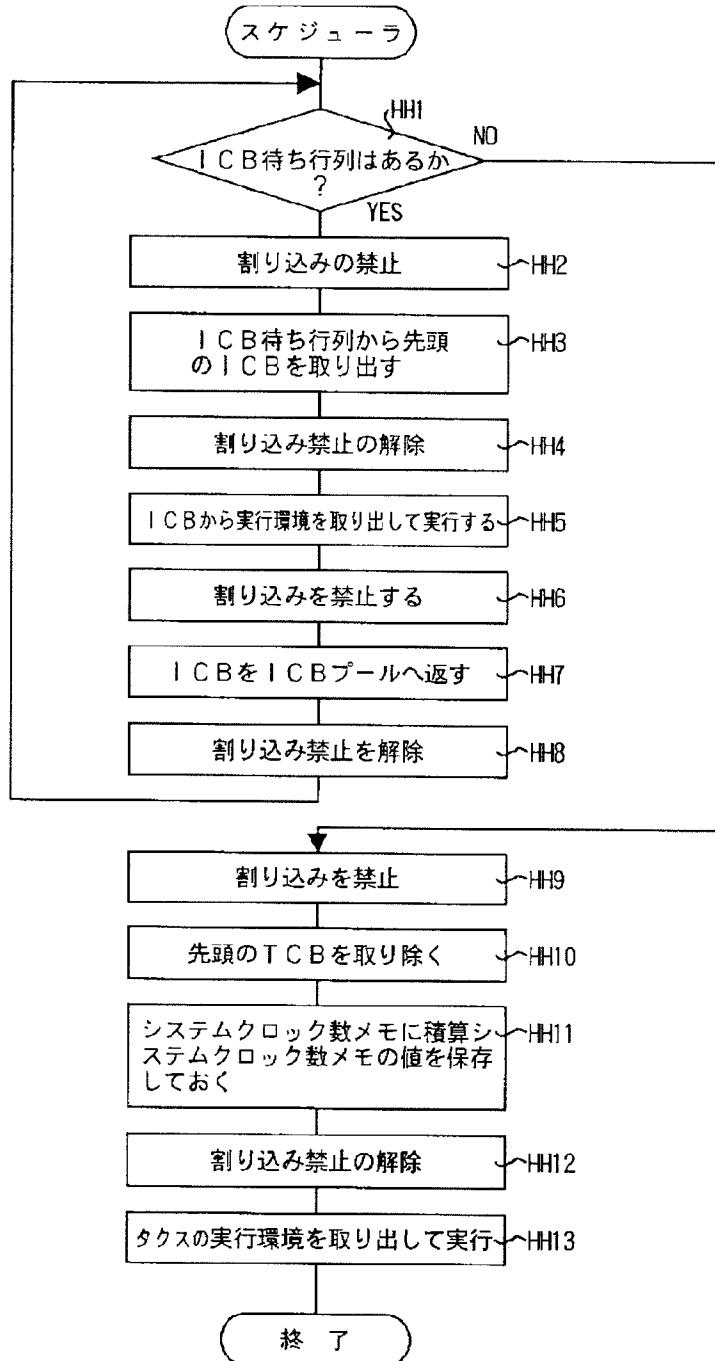
【図76】



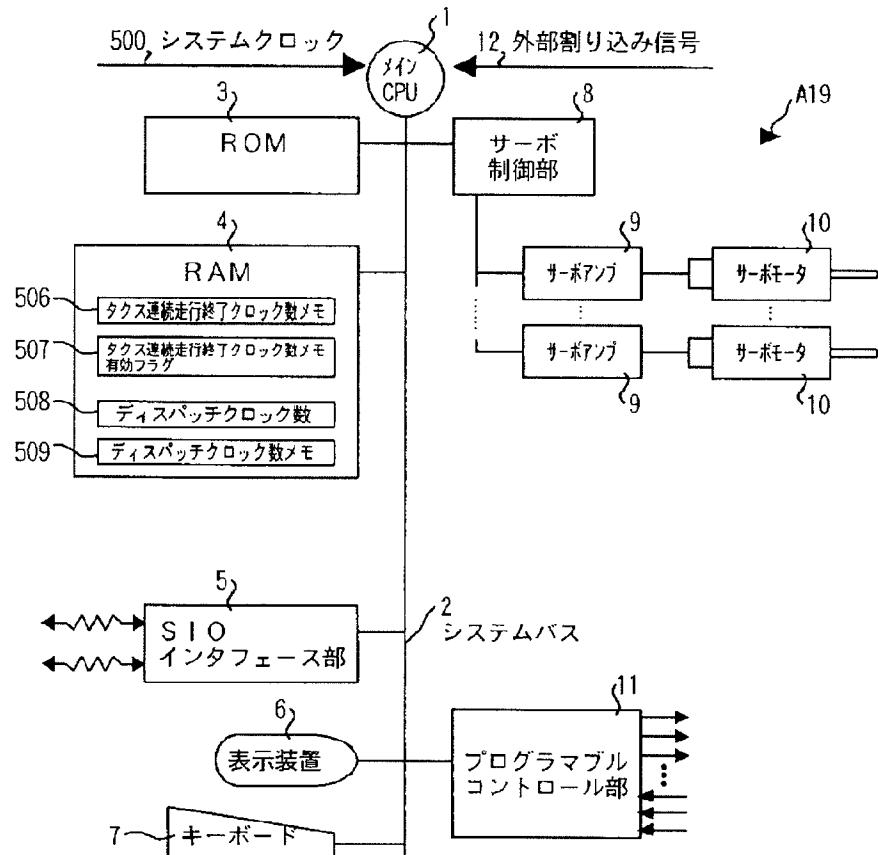
【図52】



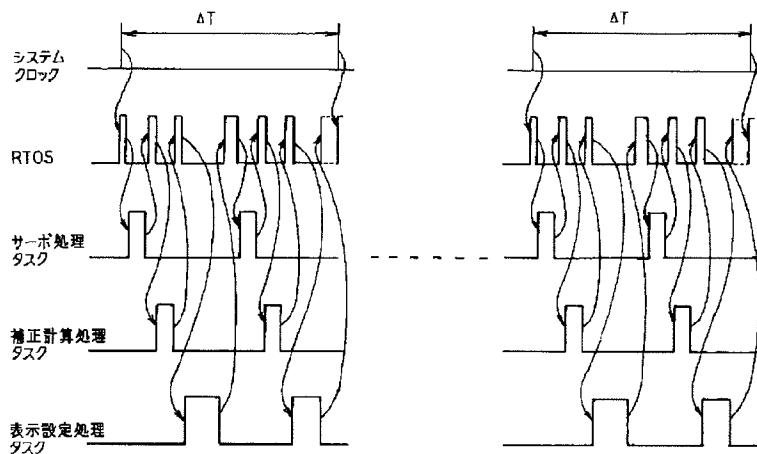
【図53】



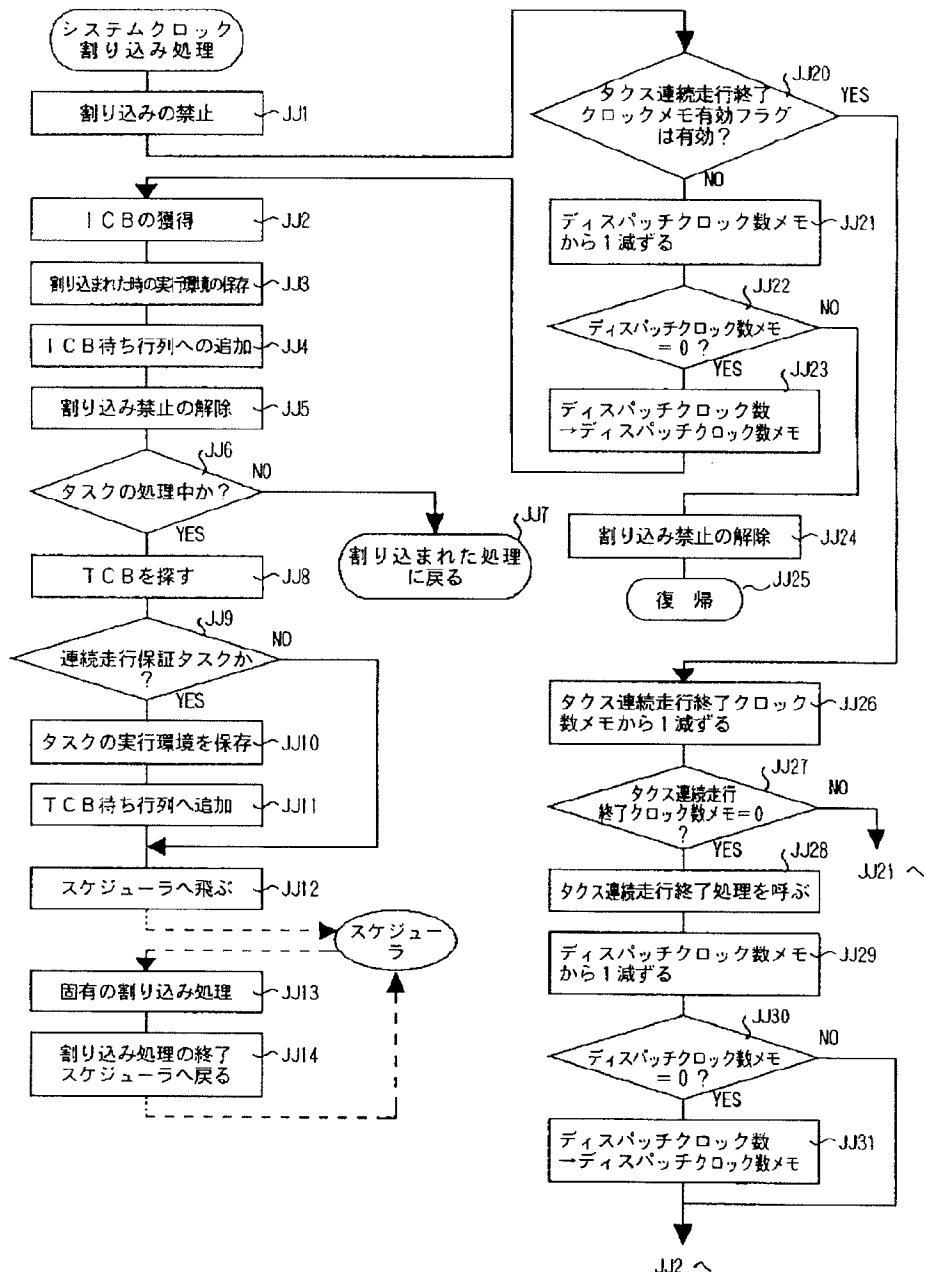
【図54】



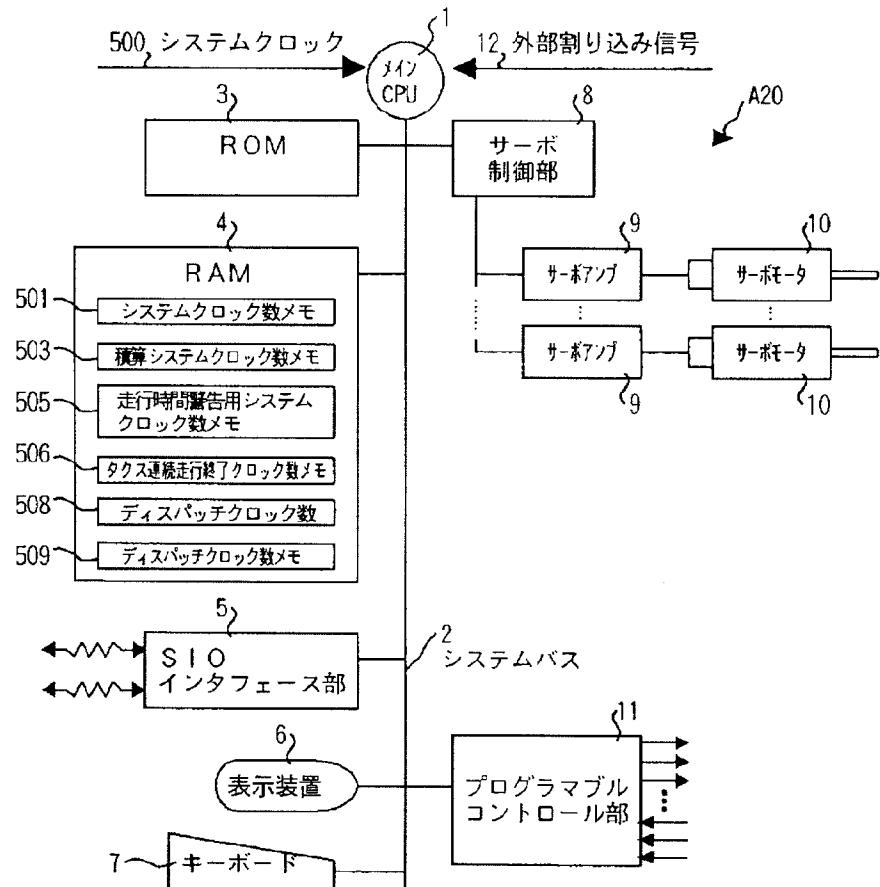
【図77】



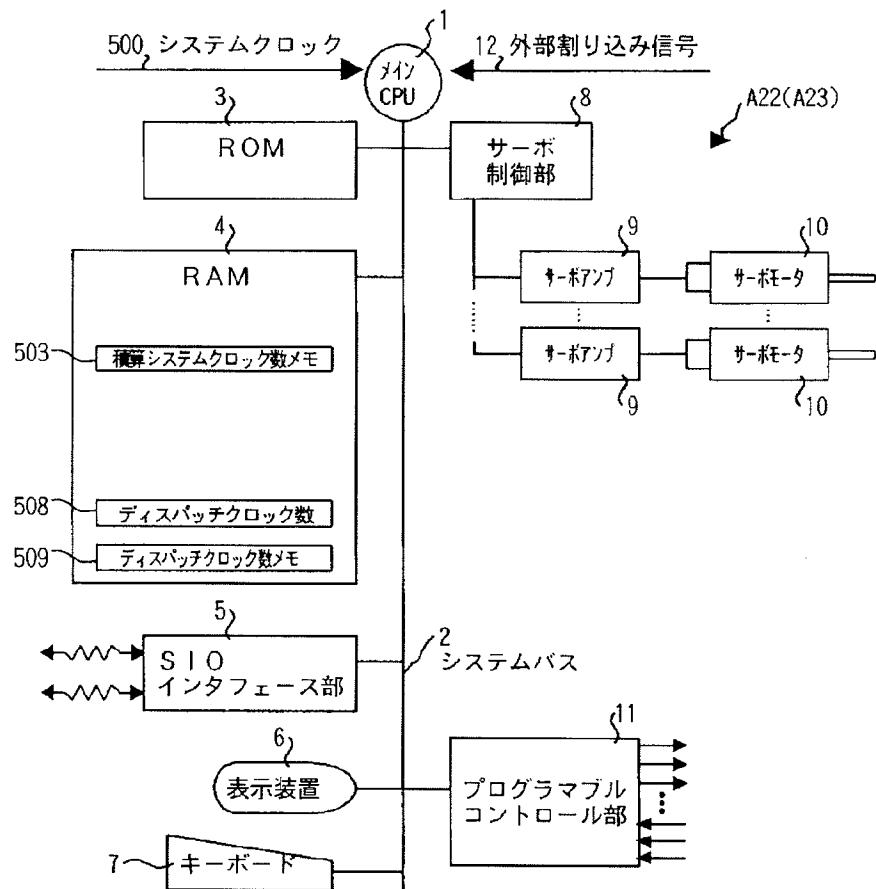
【図58】



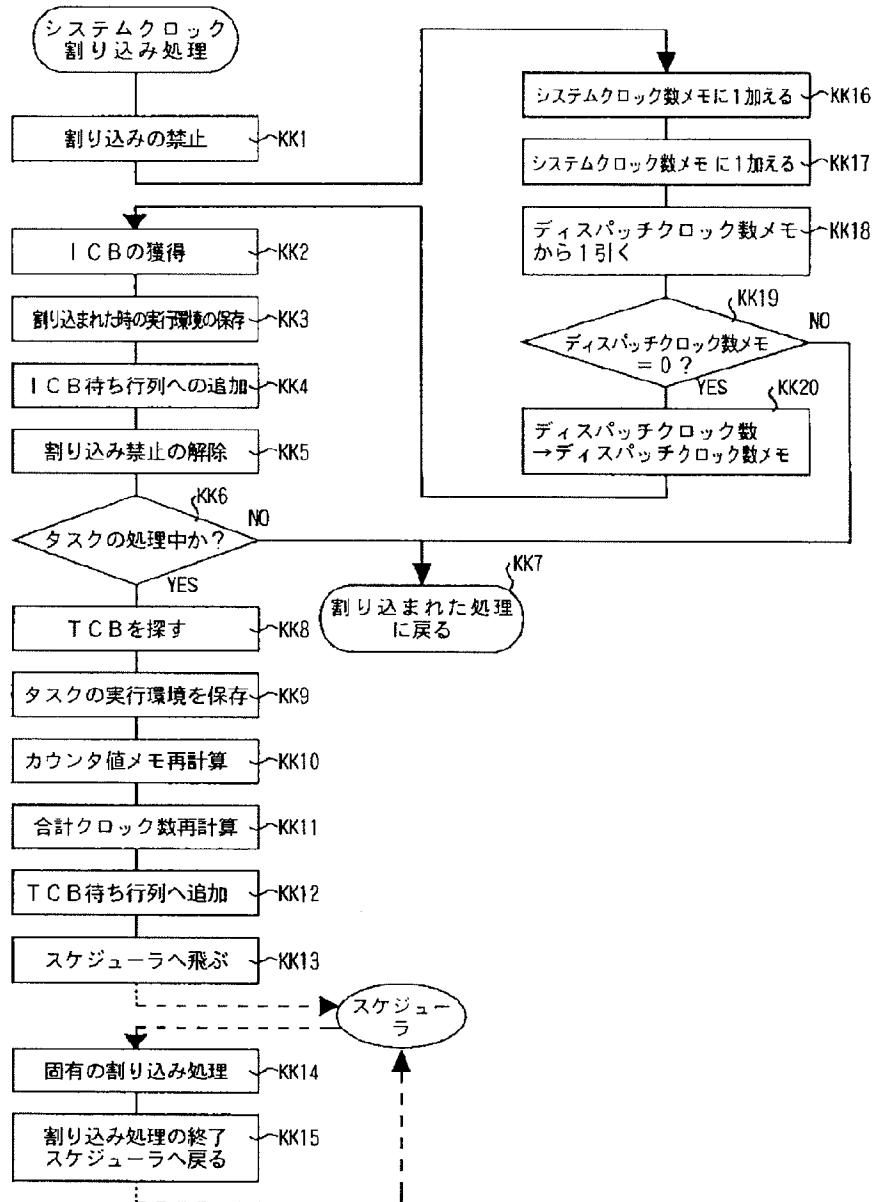
【図59】



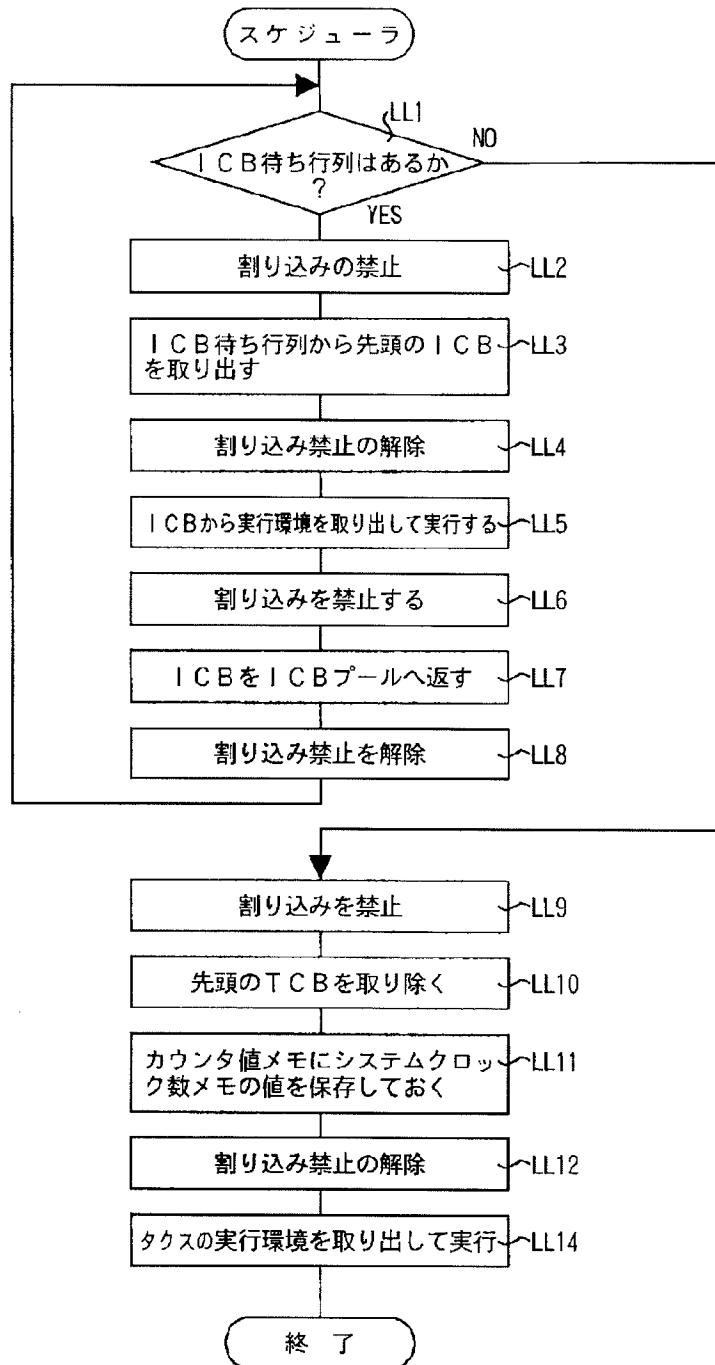
【図60】



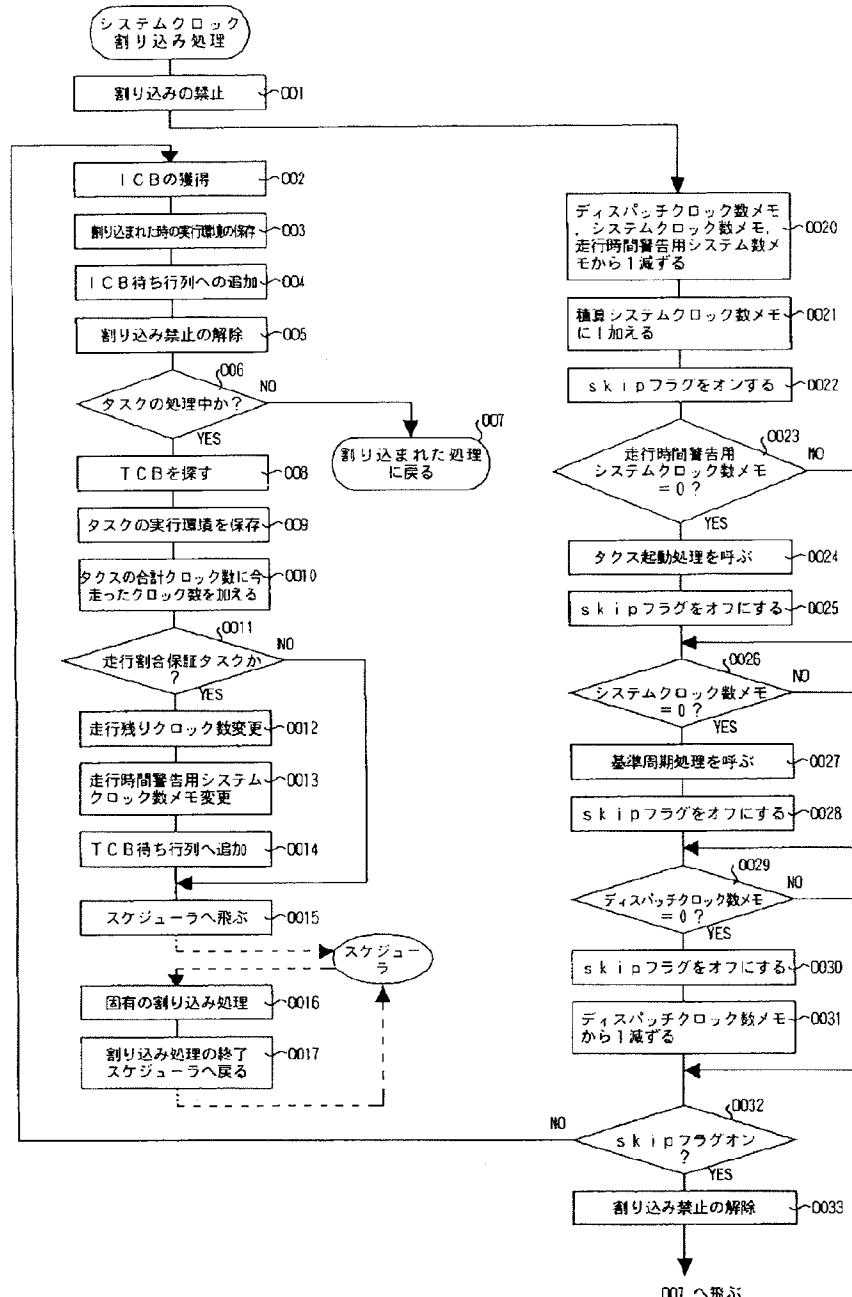
【図62】



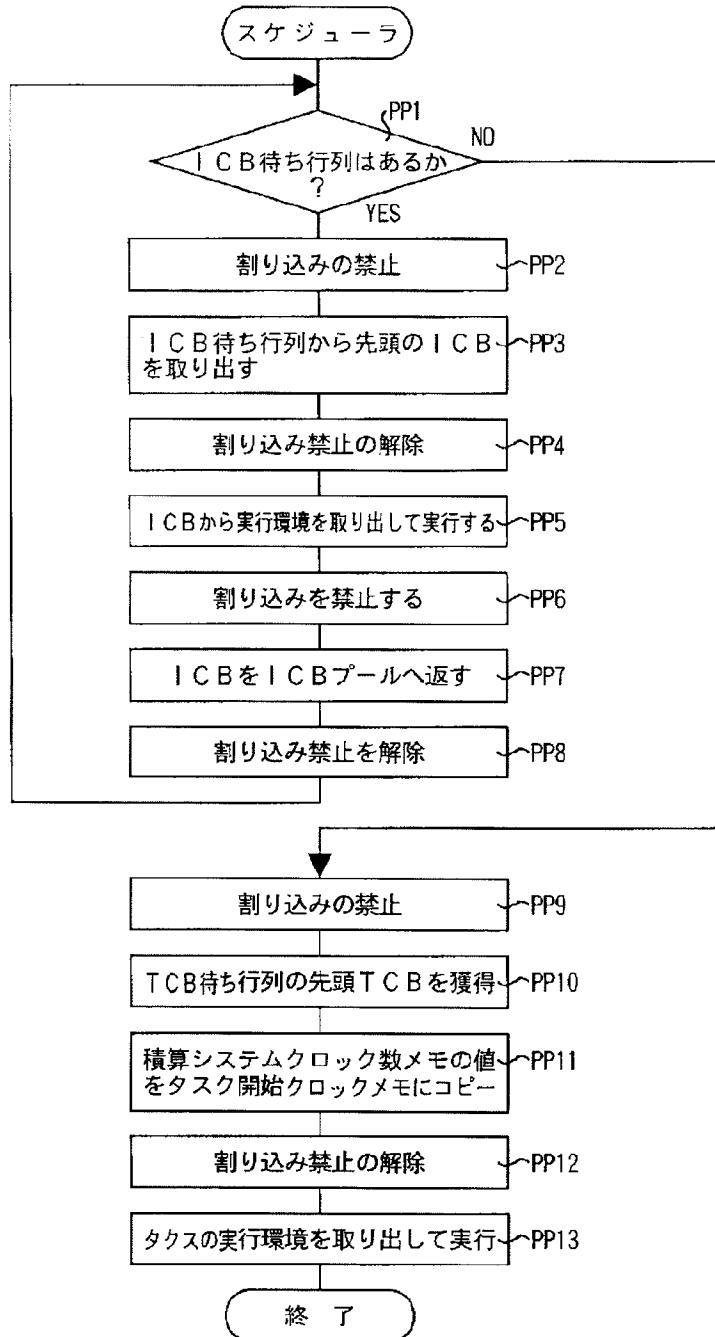
【図63】



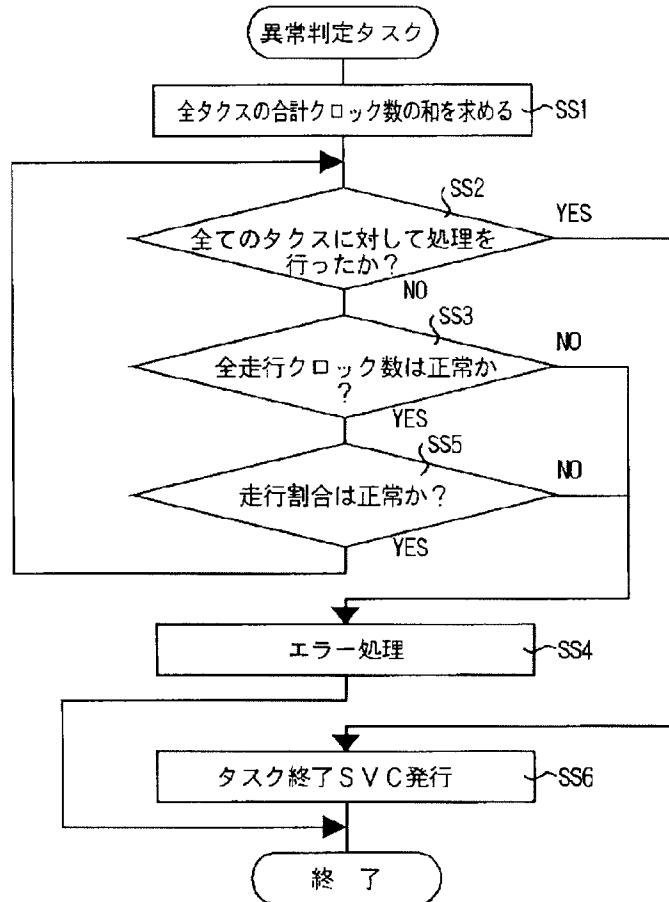
【図65】



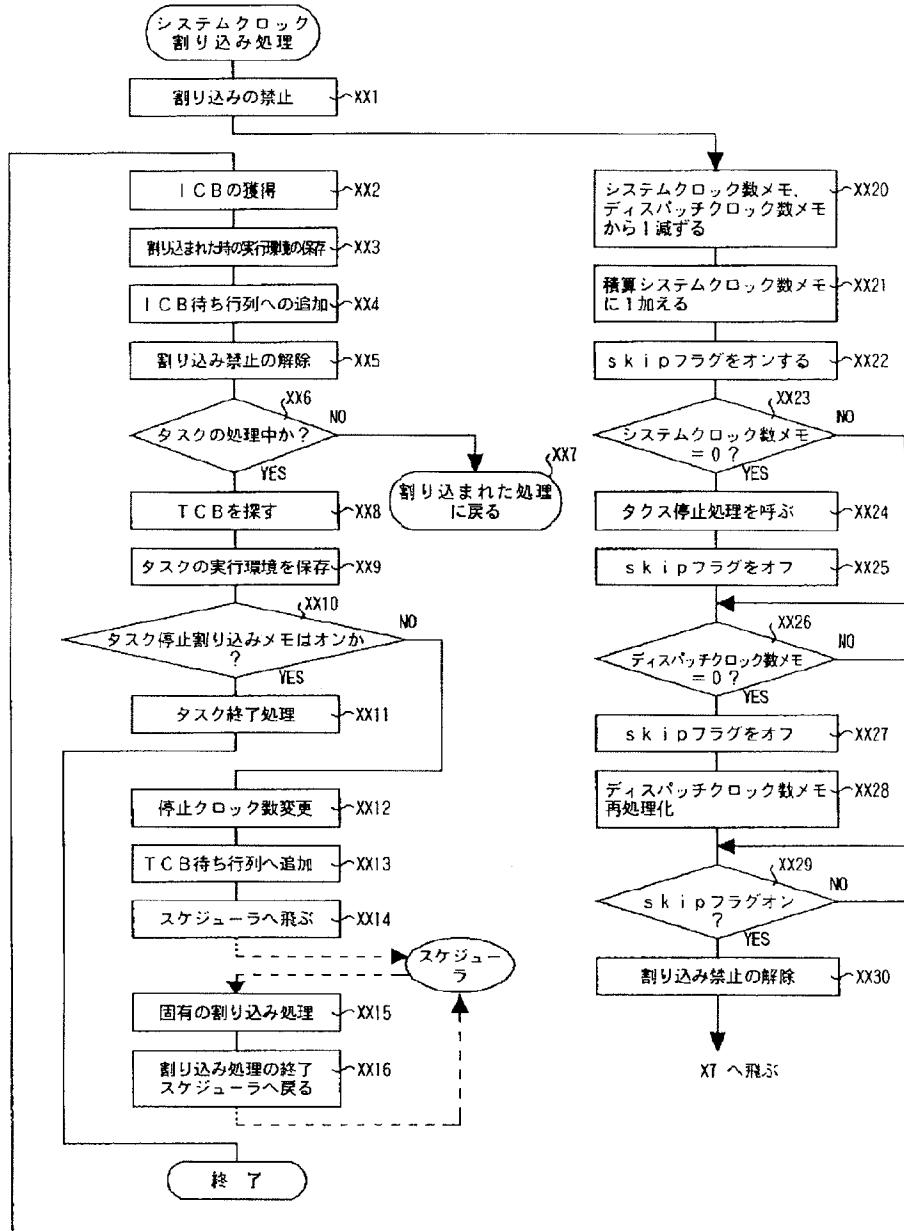
【図66】



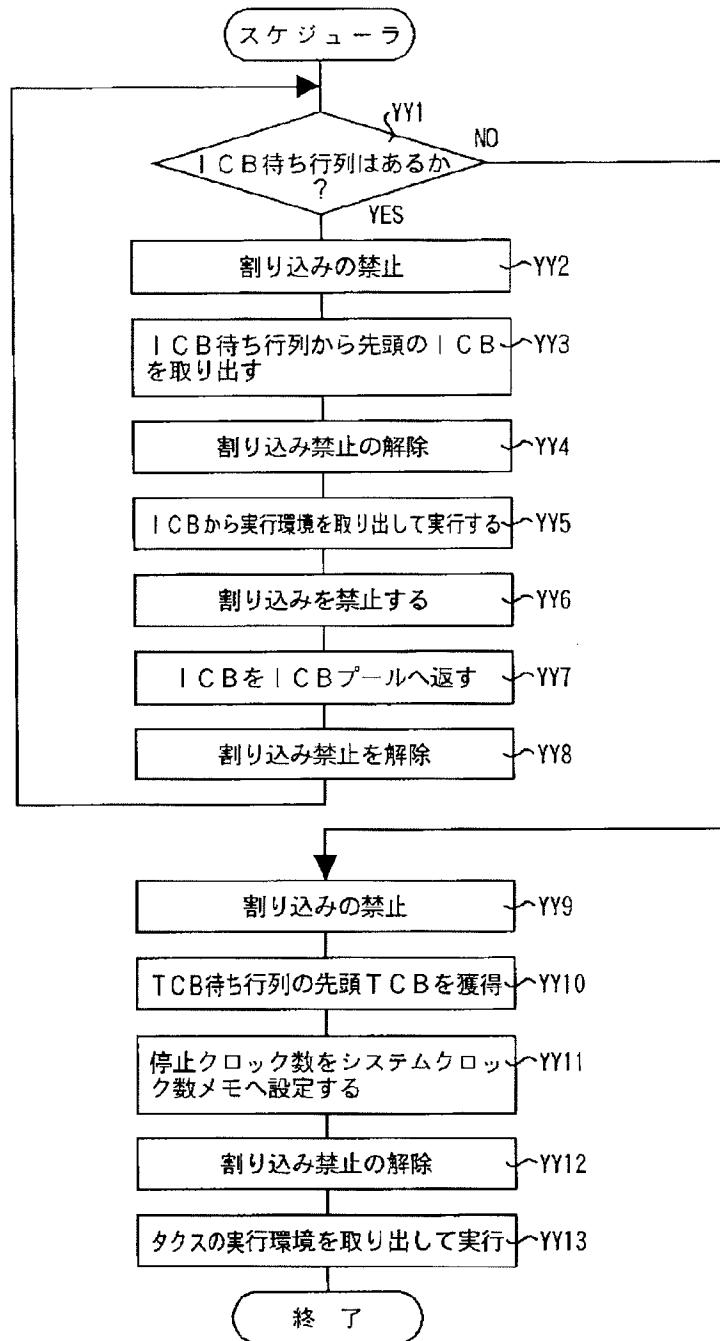
【図6.8】



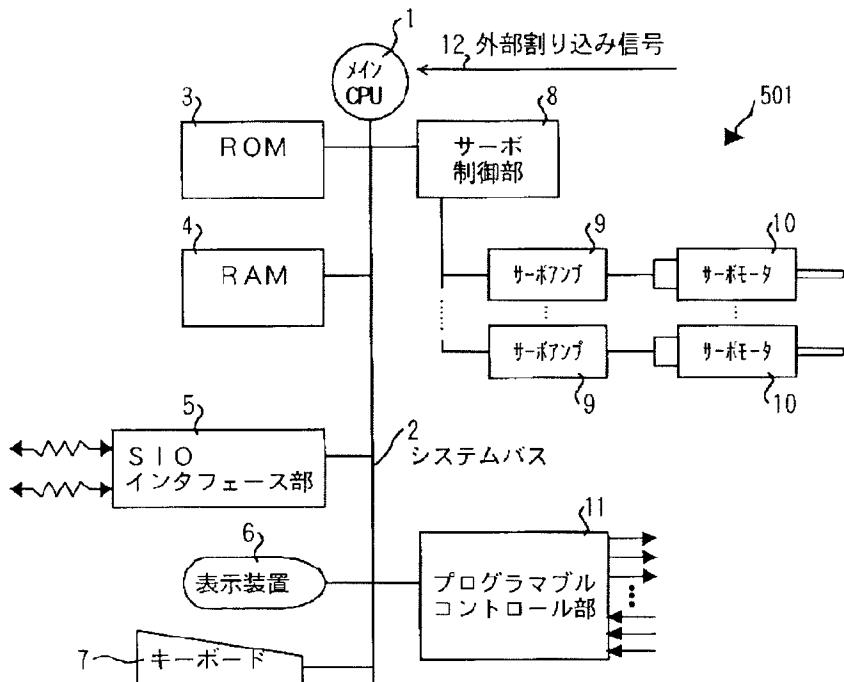
【図71】



【図72】



【図73】



## 【手続補正書】

【提出日】平成7年11月27日

## 【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】0021

【補正方法】変更

## 【補正内容】

【0021】図76は、各タスクにおける動作の時間的関係を示すタイミングムーチャートであり、図において、特に、補正計算処理タスクなどは、計算対象タスクの複雑さによって処理時間が大きく変化する。通常は、最も処理時間がかかる場合を想定してシステムクロックの周期を決定する。その結果、それ以外の処理を実行する場合には、図76に示したT1のようなメインCPU1が有効な処理をやら実行していないアイドルタイムが発生する。なお、図76において、他の処理がないときは後述のようにRTOS中のループにおいて処理が待機するものである。

## 【手続補正2】

【補正対象書類名】明細書

【補正対象項目名】0033

【補正方法】変更

## 【補正内容】

【0033】第3に、タスクの優先順位の決め方が難し

く、特に、3個以上のタスクが関係すると、いわゆるプライオリティインバージョン（優先順位の低いタスクが優先順位の高いタスクを止めてしまうような優先順位の逆転現象）が発生する可能性があるため、優先順位だけによる実行方式には限界がある。

## 【手続補正3】

【補正対象書類名】明細書

【補正対象項目名】0101

【補正方法】変更

## 【補正内容】

【0101】さらに、図1において、9はサーボ制御部8から受け取った指令を電力増幅してサーボモータや工作機械主軸（図示せず）を駆動するサーボアンプ、10は工作機械（図示せず）の加工部を制御するためのサーボモータ、11は工作機械との間でサーボ制御指令以外のデータをやり取りするためのプログラマブルコントロール部（以下、PC部という）、12はメインCPU1に入力される、システムクロック（図示せず）と外部割り込み信号を表している。ここで、システムクロックはNC装置全体を制御するために同期をとるためのクロック信号であり、また、外部割り込み信号は電源異常や非常停止などイベント（緊急の出来事）の発生をメインCPU1に通知するための信号である。

## 【手続補正4】

【補正対象書類名】明細書

【補正対象項目名】0103

【補正方法】変更

## 【補正内容】

【0103】次に、上記の構成を有するNC装置の制御ソフトウェア実行システムA1の動作について説明する。通常、システムクロックとしては60Hzから100Hz（1秒間に60回から100回）を用いる。システムクロックをこの数値以上に速くすると、その割り込み処理を実行する回数が多くなり過ぎ、NC装置の機能を実行するタスクの走行可能な時間が少なくなってしまうからである。しかし、タスクの細かい制御に対しては、これでは最小単位が大き過ぎるため、割り込みは使⽤せずに、例えは、汎用のリアルタイムクロック201を使用する。水晶発振器203によって、リアルタイムクロック201に、例えは、10MHzのクロック信号を入力すると、リアルタイムクロック201はその内部カウンタレジスタ202を1ずつ減らしていく。10MHzのクロック信号の入力の場合は0.1μs（0.000001秒）単位で測定することができる。

## 【手続補正5】

【補正対象書類名】明細書

【補正対象項目名】0138

【補正方法】変更

## 【補正内容】

【0138】また、“走行残りチック数”の欄は、1回の基準周期が終了する毎に、走行チック数の欄をコピーし、タスクが走ったチック数だけ減算していく。1回の基準周期が終る毎に、すべてのタスクの、この欄の内容が0になっていればよい。当然走行チック数の合計は基準周期よりも少なくなければならない。そのエラーチェックは、例えは、タスク状態メモに新たに登録する毎に行なえばよい。ただし、以下の説明においては、このエラーチェックについては省略する。

## 【手続補正6】

【補正対象書類名】明細書

【補正対象項目名】0149

【補正方法】変更

## 【補正内容】

【0149】F3では、全ての走行禁止フラグがオフであるTCBが、全ての走行禁止フラグがオンのTCBの前に来るよう、タスク処理の待ち行列を作り直す。すなわち、走行禁止フラグがオンのタスクの方がフラグがオフのタスクよりも前に走るようにする。F4は、割り込まれた状態がタスク処理中ならば、タスクの実行環境をTCBへ保存して、割り込みを許可し、スケジューラへ飛ぶ。反対に、割り込まれた状態がタスク処理中でなければ、割り込まれた処理に復帰する処理である。

## 【手続補正7】

## 【補正対象書類名】明細書

【補正対象項目名】0156

【補正方法】変更

## 【補正内容】

【0156】G14は、スケジューラへ飛ぶ。G15は、スケジューラから戻ってきたときの処理であり、スケジューラへ飛んだ後すぐにこの処理が実行されるとは限らず、他のTCBが処理された後の場合もあるが、いずれはここへ戻ってくる。ここからが割込みに固有の本来の割り込み処理であり、割り込み要因によって処理内容が異なる。G16は、割り込み処理の終了であり、通常のサブルーチンのリターン命令が一般的であり、このリターン命令によって、スケジューラのTCB処理部に戻る。

## 【手続補正8】

【補正対象書類名】明細書

【補正対象項目名】0191

【補正方法】変更

## 【補正内容】

【0191】以上で述べたように、実施例4に係るNC装置の制御ソフトウェア実行システムA4は、通常のプライオリティベースのスケジューリング方式しか持たないRTOSに、タスク起動周期とタスク走行割合の保証を与える制御を行なえる機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

## 【手続補正9】

【補正対象書類名】明細書

【補正対象項目名】0202

【補正方法】変更

## 【補正内容】

【0202】ここで、連続走行終了割り込みと、一般的割り込みの処理について、TCB待ち行列への操作を中心まとめる。すなわち、

① タスク起動割り込みは、この割り込みによって起動するタスクのTCBを、TCB待ち行列の先頭に挿入する、

② 連続走行終了割り込みは、実行中のタスクの実行中フラグをオフして他のタスクの走行を許可する、

③ 一般的の割り込みは、連続走行を保証するタスクの場合には、TCB待ち行列のディスパッチを実行しない、がある。

## 【手続補正10】

【補正対象書類名】明細書

【補正対象項目名】0310

【補正方法】変更

## 【補正内容】

【0310】本実施例に係るNC装置の制御ソフトウェア実行システムA15は、タスクを起動するたびに、もし指定されれば、タスク停止テーブル400（図3

9参照)の停止チック数を、リアルタイムクロック201(図9参照)の内部カウンタレジスタ204に設定する。そして、そのタスク停止割り込みが発生すれば、当該タスクを強制的に停止させる。

【手続補正11】

【補正対象書類名】明細書

【補正対象項目名】0311

【補正方法】変更

【補正内容】

【0311】もし、タスク停止割り込みが発生する前に、他のタスクによってメインCPU1を占有された場合は、リアルタイムクロック201(図9参照)の内部カウンタレジスタは202(図9参照)と、タスク開始チックメモ210(図10参照)から、当該タスクの走行チック数を求め、その分をタスク停止テーブル400(図39参照)の停止チック数から減ずる。このようにして、指定タスクの最大走行チック数を制限することができる。

【手続補正12】

【補正対象書類名】明細書

【補正対象項目名】0318

【補正方法】変更

【補正内容】

【0318】X12は、タスク停止割り込みメモ411(図39参照)がオフのときの処理であり、リアルタイムクロック201(図9参照)の内部カウンタレジスタ202(図9参照)と、タスク開始チックメモ210(図10参照)との差から、当該タスクの走行チック数を求め、タスク停止テーブル400(図39参照)の停止チック数から減ずる。X13は、X9において作成したTCBを、スケジューラが実行するTCB待ち行列にプライオリティの順に追加する。X14は、スケジューラへ飛ぶ。

【手続補正13】

【補正対象書類名】明細書

【補正対象項目名】0323

【補正方法】変更

【補正内容】

【0323】Y9は、割り込みを禁止する。Y10は、割り込み処理の待ち行列のリストから、先頭のTCBを獲得する。Y11は、当該TCBのタスクが、タスク停止テーブル400(図39参照)の停止チック数が、0以上のタスクである場合は、当該タスクのタスク停止テーブル400(図39参照)の停止チック数の値を、リアルタイムクロック201(図9参照)の内部カウンタレジスタ204に設定する。この結果、設定した数だけのチック数が経過すると、タスク停止割り込みが発生する。Y12は、割り込み禁止を解除する。Y13は、Y10においてタスク処理の待ち行列(リスト)から獲得した、TCBからX9(図41参照)において格納した

タスクの実行環境を取り出して実行する。その処理へ飛んでいって、ここへ戻ってこない。

【手続補正14】

【補正対象書類名】明細書

【補正対象項目名】0348

【補正方法】変更

【補正内容】

【0348】その結果、BB8において、全ての走行待ちブロック52を調べ終っていないと判定した場合には、BB9に処理が移行する。BB9では、走行待ちブロック52が、走行待ちリスト51中にあったと判断した場合の処理であり、走行待ちリスト51の中から、T1の示す走行待ちブロック52の、走行予定クロック数を取り出してLTに加える。

【手続補正15】

【補正対象書類名】明細書

【補正対象項目名】0384

【補正方法】変更

【補正内容】

【0384】図50は、基準周期リセット処理の手順について説明したフローチャートである。基準周期リセット処理は、基準周期毎のシステムおよび全タスクの再初期化処理と考えられる。EE2では、タスク走行状態メモ512の走行残りクロック数の合計を求める。EE3では、その合計が0以上か否かを判定し、0以上ではないと判定した場合には、EE4において何らかのエラー処理を行なうが、通常は発生しないのでここでは、その説明を省略する。

【手続補正16】

【補正対象書類名】明細書

【補正対象項目名】0408

【補正方法】変更

【補正内容】

【0408】【実施例19】次に、実施例19の構成について説明する。実施例19に係るNC装置の制御ソフトウェア実行システムA19の全体構成を示す図54の内容にあっては、実施例18において示した図48の内容とほぼ同様である。以下、構成について詳細に説明する。この制御ソフトウェア実行システムA19の1~12, 500, 508, 509は、上記実施例17の図44に示した構成と同様であるので、その説明を省略する。

【手続補正17】

【補正対象書類名】明細書

【補正対象項目名】0412

【補正方法】変更

【補正内容】

【0412】このテーブルはタスク起動のSVCにおいて、そのパラメータとして指定することにすれば必要はなくなるが、以下の説明の都合上、ここではこのテーブ

ルを用いることにする。

【手続補正 18】

【補正対象書類名】明細書

【補正対象項目名】0417

【補正方法】変更

【補正内容】

【0417】JJ21は、タスク連続走行終了クロック数メモ有効フラグ507が有効でなかった場合の処理で、ディスパッチクロック数メモ509から1を減算する。JJ22は、上記ディスパッチクロック数メモ509が0になったか否かを判定する処理である。JJ23は、ディスパッチクロック数メモ509の値をディスパッチクロック数508で再初期化する処理である。JJ24は、割り込み禁止の解除処理である。JJ25は、割り込まれた処理に直接戻す処理である。

【手続補正 19】

【補正対象書類名】明細書

【補正対象項目名】0448

【補正方法】変更

【補正内容】

【0448】次に、実施例21に係るNC装置の制御ソフトウェア実行システムA21の動作について説明する。実施例20と同様に、各タスクは予め、あるいはそのタスクが開始させられるときにSVCによってその実行周期を登録する。

【手続補正 20】

【補正対象書類名】明細書

【補正対象項目名】0485

【補正方法】変更

【補正内容】

【0485】次に、本実施例に係る、NC装置の制御ネットワーク実行システムA23の動作について説明する。本実施例では、制御ソフトウェア実行システムA23(図60参照)のRTOSは、予め、例えば、システムを稼働したとき等に、システムクロック数メモ501を0で初期化しておく。

【手続補正 21】

【補正対象書類名】明細書

【補正対象項目名】0509

【補正方法】変更

【補正内容】

【0509】OO2は、未使用のICBをICBプール(未使用ICBを保持しておく場所)から1つ獲得する。OO3は、ICBに、この割り込み処理(自分自身の)の、実行環境を格納する。この中には次に実行する命令のアドレスとしてOO8の処理を先頭アドレスをICBに格納することも含まれる。

【手続補正 22】

【補正対象書類名】明細書

【補正対象項目名】0511

【補正方法】変更

【補正内容】

【0511】OO7は割り込まれた処理が割り込み処理またはRTOSを実行中のときで、このときは割り込まれた処理に直接戻る。OO8は割り込まれた処理がタスクの処理を実行中のときで、このときはまず、割り込まれたタスクのTCBを探し出し、OO9で当該TCBへ割り込まれたタスクの実行環境を格納する。

【手続補正 23】

【補正対象書類名】明細書

【補正対象項目名】0512

【補正方法】変更

【補正内容】

【0512】OO10は積算システムクロック数メモ503の値から、基準周期511に保存してある値を減算する。この結果が当該タスクが今回連続して走ったクロック数である。これをタスク走行時間テーブル531の当該タスクの合計クロック数の欄に加える。

【手続補正 24】

【補正対象書類名】明細書

【補正対象項目名】0520

【補正方法】変更

【補正内容】

【0520】PP6はOO17(図65参照)から戻ってくるところであり、再び割り込みを禁止する。PP7はPP3で取り除いたICBをICBプールへ戻す。PP8は割り込み禁止を解除してからPP1へ戻る。後は割り込み処理待ち行列がある間はPP1からPP7の処理を繰り返す。

【手続補正 25】

【補正対象書類名】明細書

【補正対象項目名】0547

【補正方法】変更

【補正内容】

【0547】以上で説明した通り、実施例27に係るNC装置の制御ソフトウェア実行システムA27にあっては、NC装置の機能を実現しているタスクに異常が発生したときにそれを検知する機能を備えている。

【手続補正 26】

【補正対象書類名】明細書

【補正対象項目名】0555

【補正方法】変更

【補正内容】

【0555】次に、NC装置の制御ソフトウェア実行システムA28の動作について説明する。最初にタスク停止処理について説明する。図70は、タスク停止処理の手順について説明したフローチャートである。

【手続補正 27】

【補正対象書類名】明細書

【補正対象項目名】0569

【補正方法】変更

【補正内容】

【0569】図72を用いて、本実施例に係るNC装置の制御ソフトウェア実行システムA28のスケジューラの動作を説明する。以下の、説明の都合上、本実施例においては、優先順位が最低で、外部に対して何も処理を行わないアイドルタスクが常に走っているものとする。

【手続補正28】

【補正対象書類名】明細書

【補正対象項目名】0575

【補正方法】変更

【補正内容】

【0575】以上、説明したように、実施例28に係るNC装置の制御ソフトウェア実行システムA28は、通常のプライオリティーベースのスケジューリング方式しか持たないRTOSに、タスク走行時間の制限を行える機能を追加した方式であり、この方式に基づいてNC装置の制御ソフトウェアを実行するものである。

【手続補正29】

【補正対象書類名】明細書

【補正対象項目名】0582

【補正方法】変更

【補正内容】

【0582】(各実施例の効果)以上のように、この発明の各実施例に係るNC装置の制御ソフトウェア実行システムによれば、NC装置の機能を分担して実現している、タスクの制御をきめ細かく実行でき、各タスクの処理時間を最適化することができるため、無駄時間を少なくて、全体の処理時間を早めることができる。また、NC装置の制御ネットワーク実行システムの異常を、システム自体が判定できる手段を備えているため、異常事態が発生したときの回避処理も容易となる。

【手続補正30】

【補正対象書類名】明細書

【補正対象項目名】0599

【補正方法】変更

【補正内容】

【0599】また、この発明に係るNC装置の制御ソフトウェア実行システムの制御方法は、NC装置の機能を実現するタスクの時間による制御を小さな時間単位で行えるようになり、さらにタスクを起動する周期を簡単に指定できるため、タスクに割り当てた時間の無駄を少な

くでき、さらに、タスクをその走行時間で制御できるため、システムの構築が容易になる。

【手続補正31】

【補正対象書類名】明細書

【補正対象項目名】0600

【補正方法】変更

【補正内容】

【0600】また、次の発明に係る制御ソフトウェア実行システムの制御方法は、NC装置機能を実現するタスクを走行時間割合による制御を実行することができるため、タスクにわりあてた時間の無駄を少なくでき、さらに、タスクをその走行時間において制御できるため、システムの構築が容易になる。

【手続補正32】

【補正対象書類名】明細書

【補正対象項目名】図45

【補正方法】変更

【補正内容】

【図45】実施例17に係るNC装置の制御ソフトウェア実行システムがタスク走行待ちリストを作成するときの処理手順を示すフローチャートである。

【手続補正33】

【補正対象書類名】明細書

【補正対象項目名】図49

【補正方法】変更

【補正内容】

【図49】実施例18に係るNC装置の制御ソフトウェア実行システムにおいて必要なデータ構造を示す説明図である。

【手続補正34】

【補正対象書類名】明細書

【補正対象項目名】図54

【補正方法】変更

【補正内容】

【図54】実施例19に係るNC装置の制御ソフトウェア実行システムの全体構成を示すブロック図である。

【手続補正35】

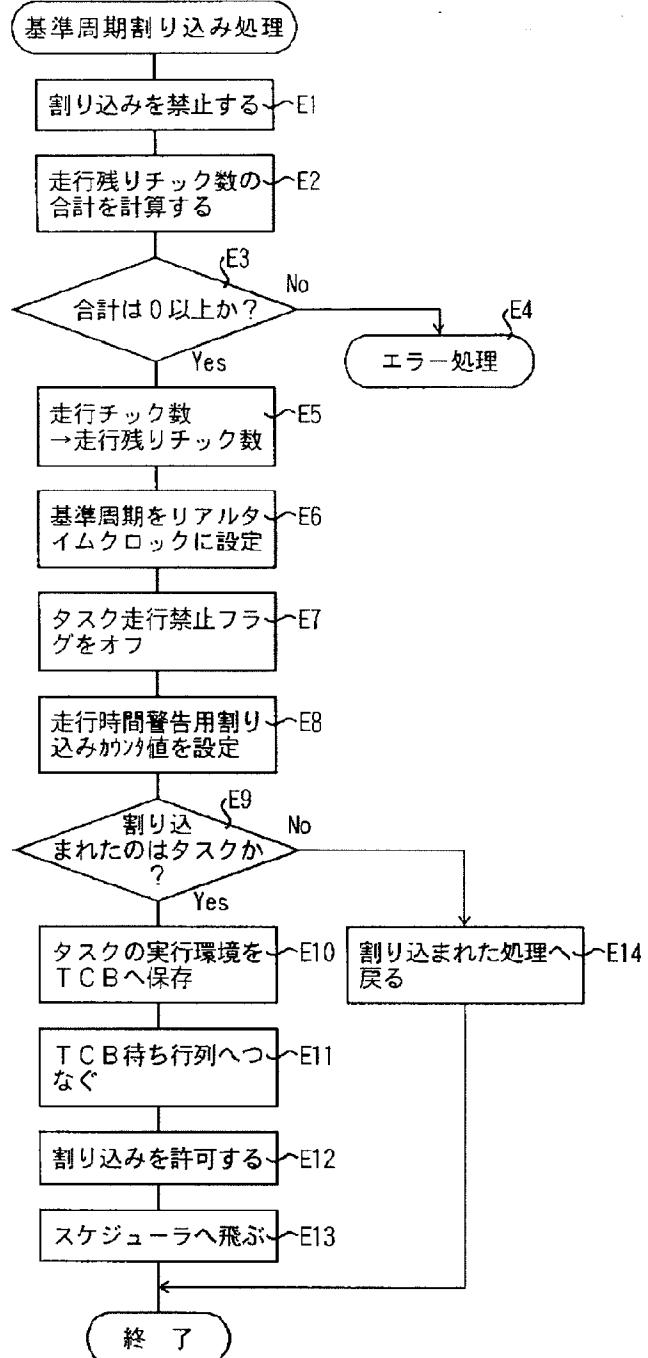
【補正対象書類名】図面

【補正対象項目名】図11

【補正方法】変更

【補正内容】

【図11】



【手続補正36】

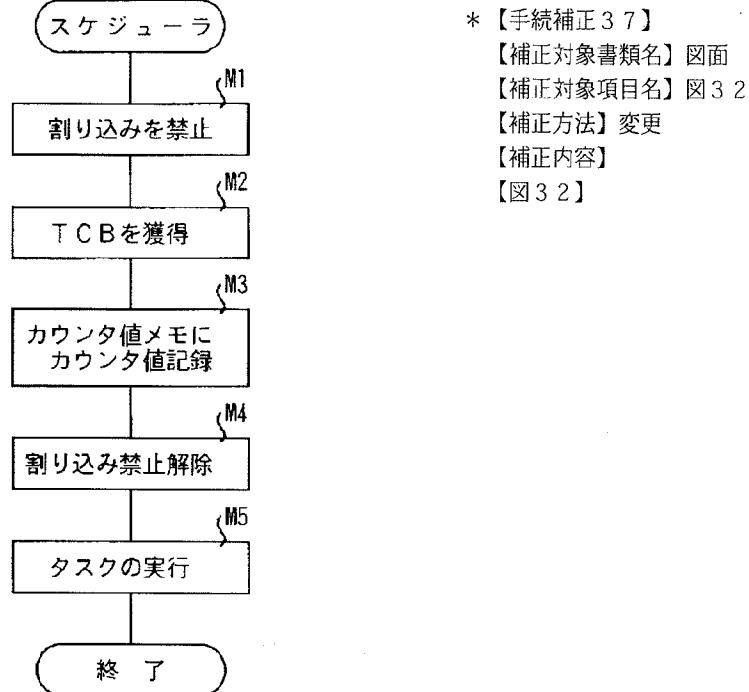
【補正対象書類名】図面

【補正対象項目名】図25

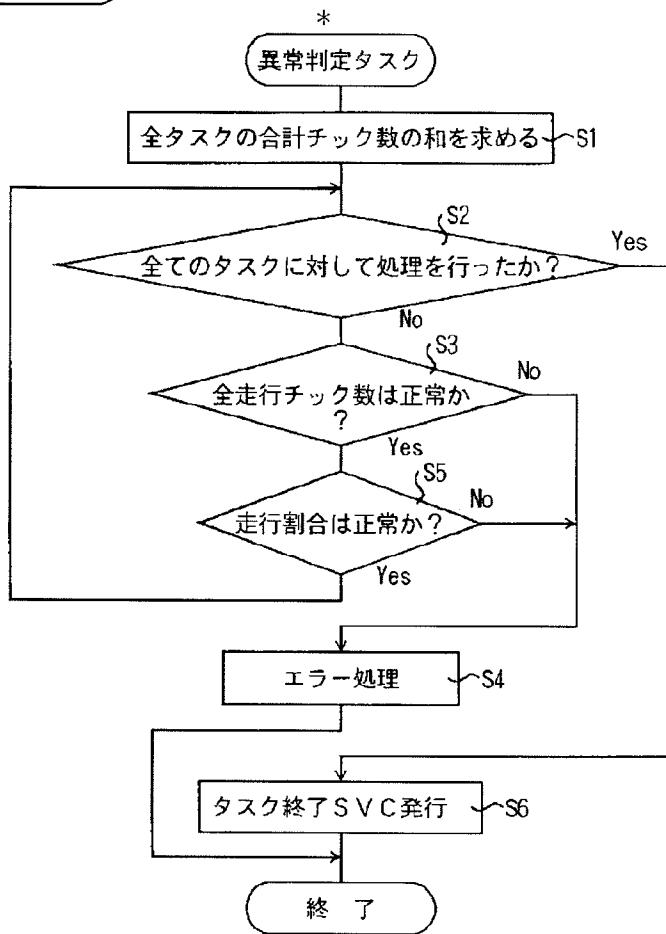
【補正方法】変更

【補正内容】

【図25】



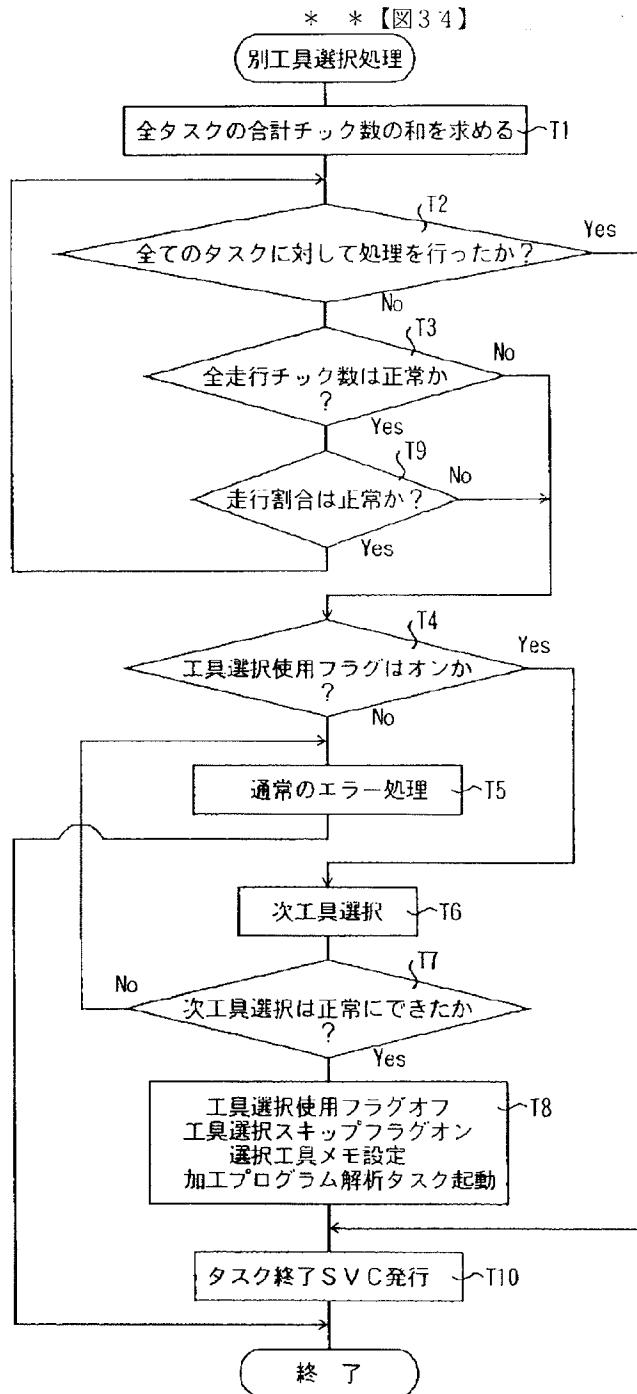
\* 【手続補正37】  
 【補正対象書類名】図面  
 【補正対象項目名】図32  
 【補正方法】変更  
 【補正内容】  
 【図32】



【手続補正38】  
 【補正対象書類名】図面

【補正対象項目名】図34  
 【補正方法】変更

【補正内容】



【手続補正39】

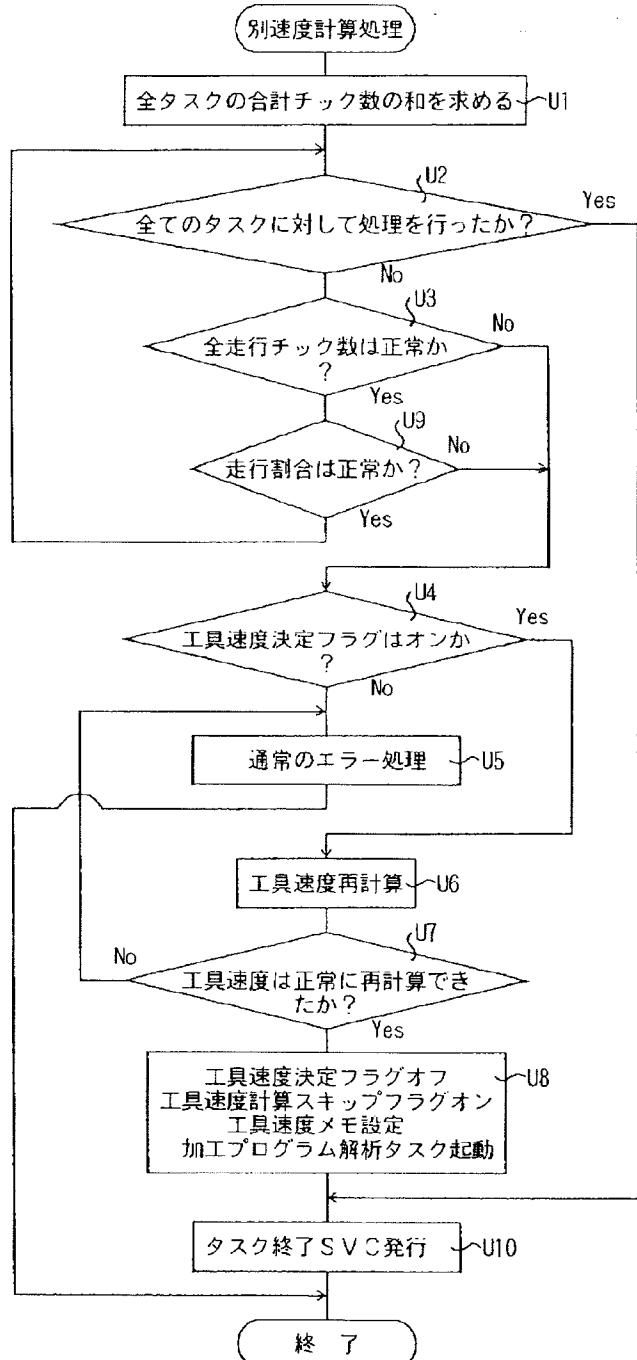
【補正対象書類名】図面

【補正対象項目名】図36

【補正方法】変更

【補正内容】

【図36】



【手続補正40】

【補正対象書類名】図面

【補正対象項目名】図37

【補正方法】変更

【補正内容】

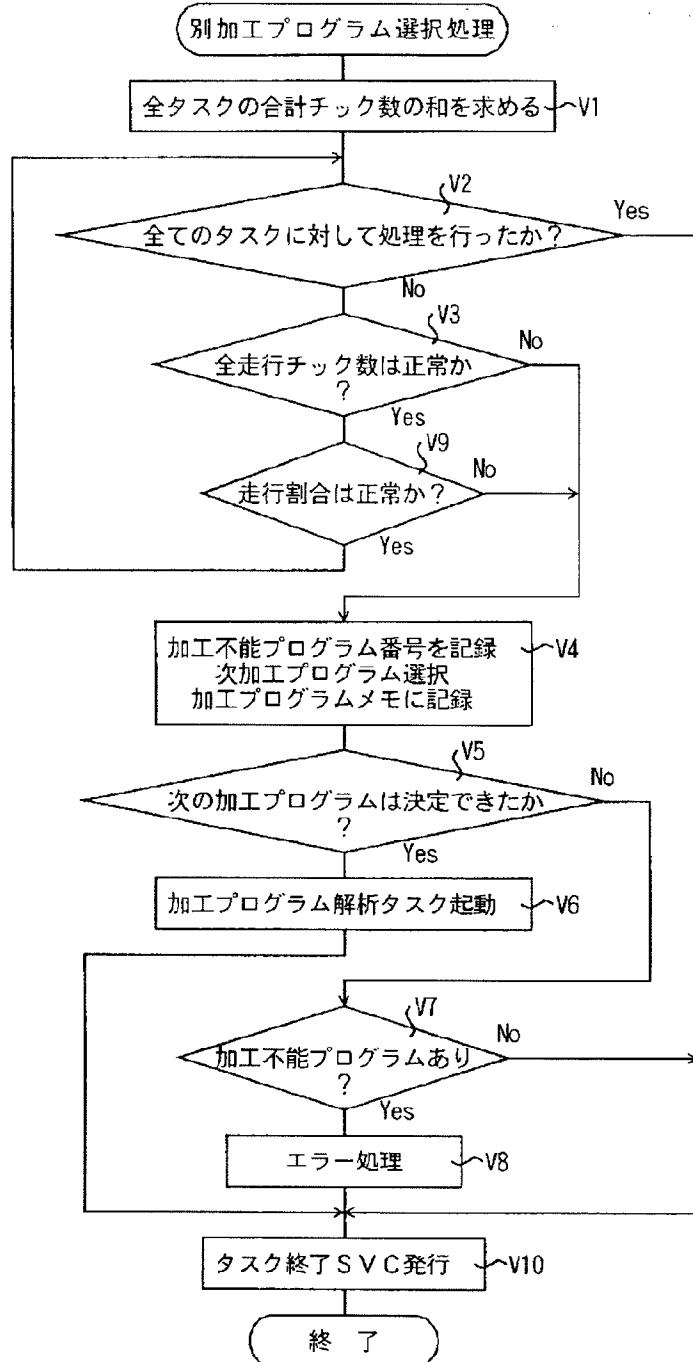
【図37】

380 加工プログラム番号メモ

381 加工プログラムスケジューリングテーブル	
加工プログラムメモ	回数

382 加工不能プログラムメモ

加工プログラム番号



【手続補正42】

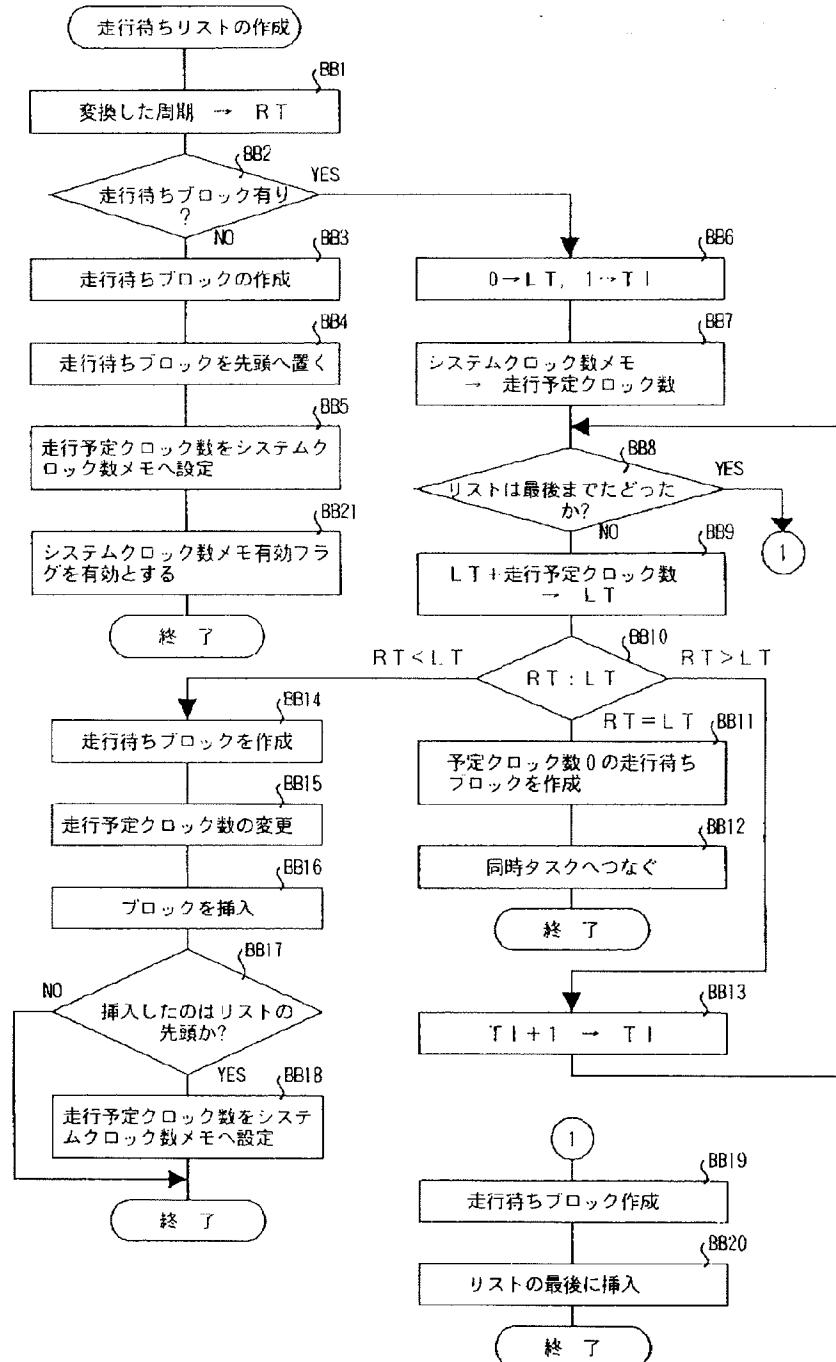
【補正対象書類名】図面

【補正対象項目名】図45

【補正方法】変更

【補正内容】

【図45】



【手続補正43】

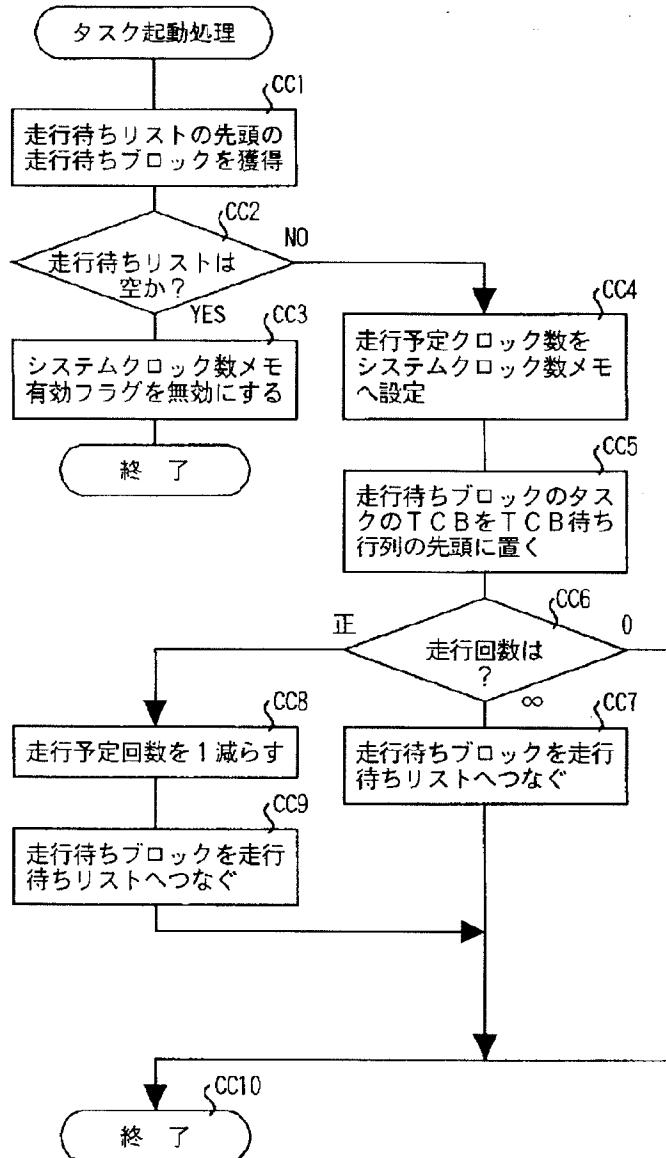
【補正対象書類名】図面

【補正対象項目名】図46

【補正方法】変更

【補正内容】

【図46】



【手続補正44】

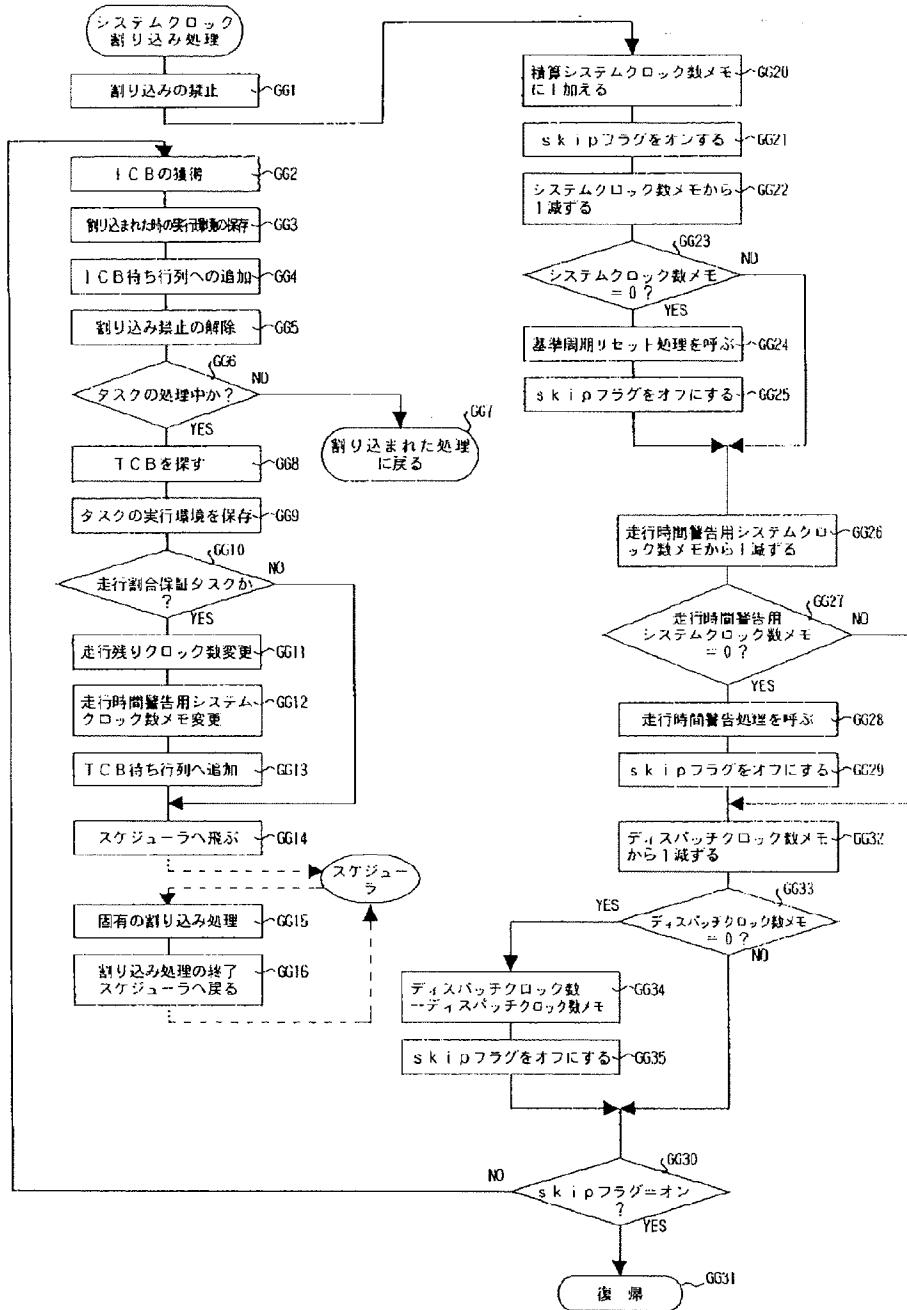
【補正対象書類名】図面

【補正対象項目名】図52

【補正方法】変更

【補正内容】

【図52】



【手続補正45】

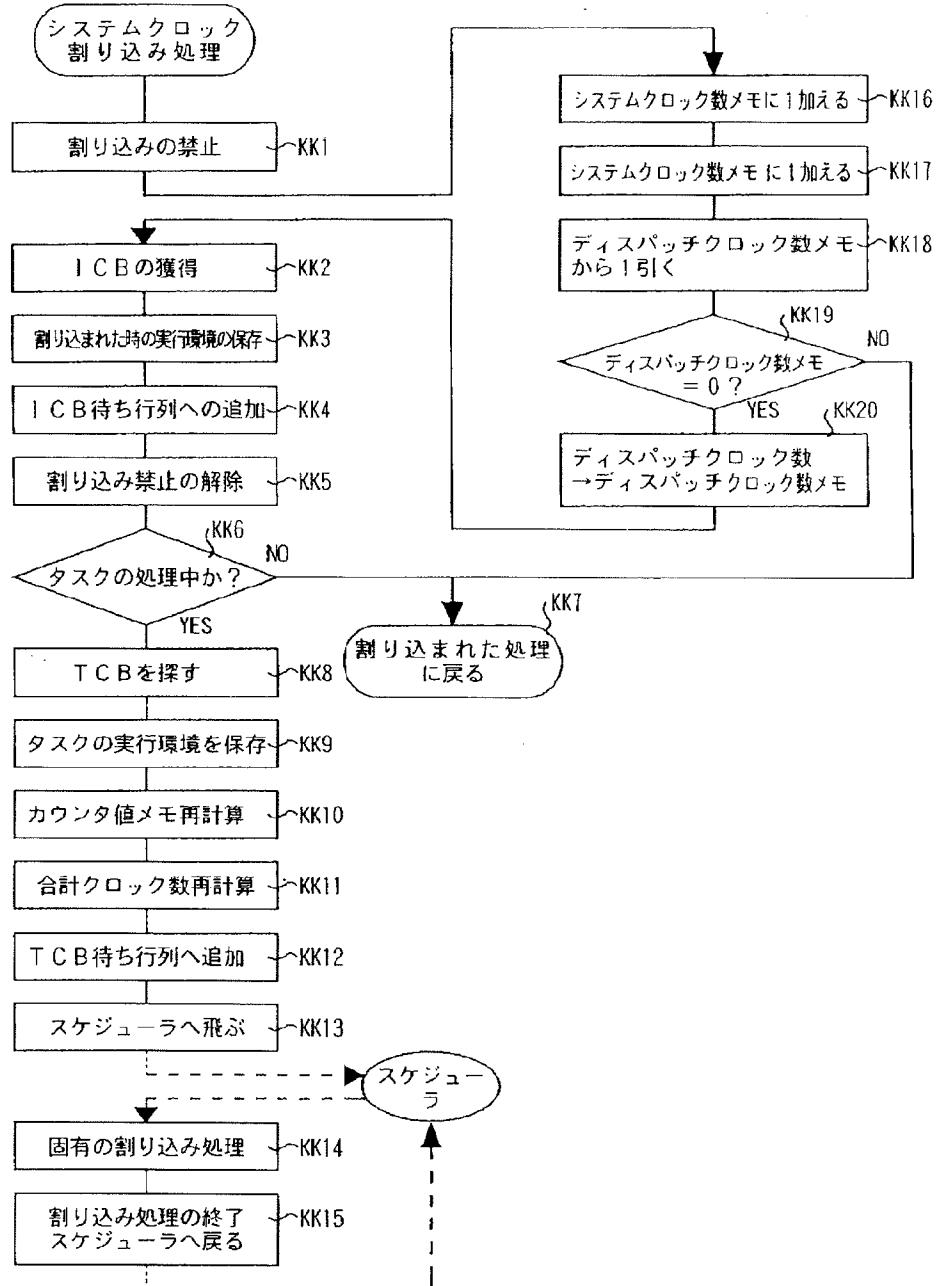
【補正対象書類名】図面

【補正対象項目名】図62

【補正方法】変更

【補正内容】

【図62】



【手続補正46】

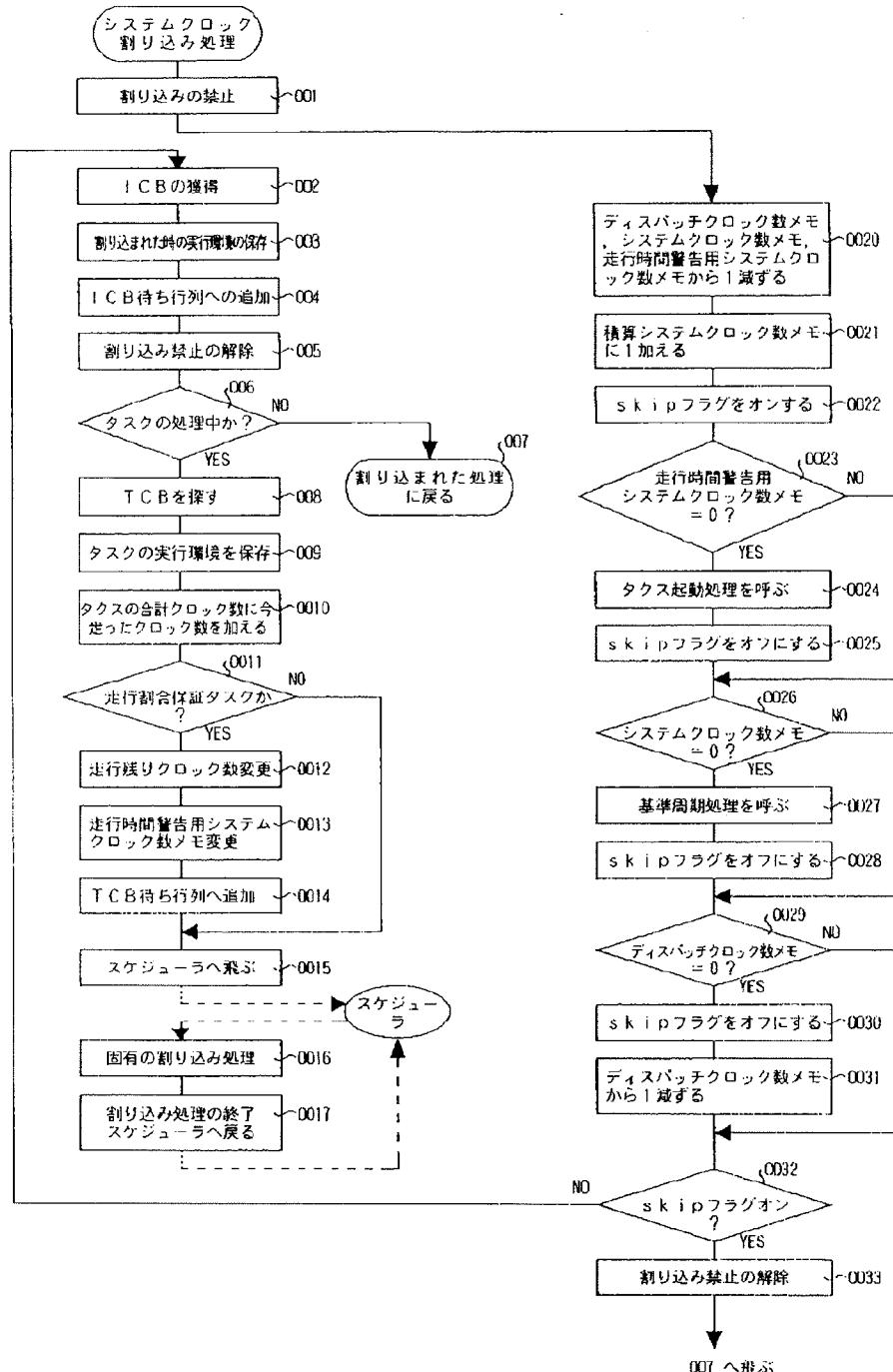
【補正対象書類名】図面

【補正対象項目名】図65

【補正方法】変更

【補正内容】

【図65】



【手続補正47】

【補正対象書類名】図面

【補正対象項目名】図67

【補正方法】変更

【補正内容】

【図67】

550 タスク走行標準データ

タスク名	走行割合	全走行クロック数

## 【手続補正48】

【補正対象書類名】図面

【補正対象項目名】図71

【補正方法】変更

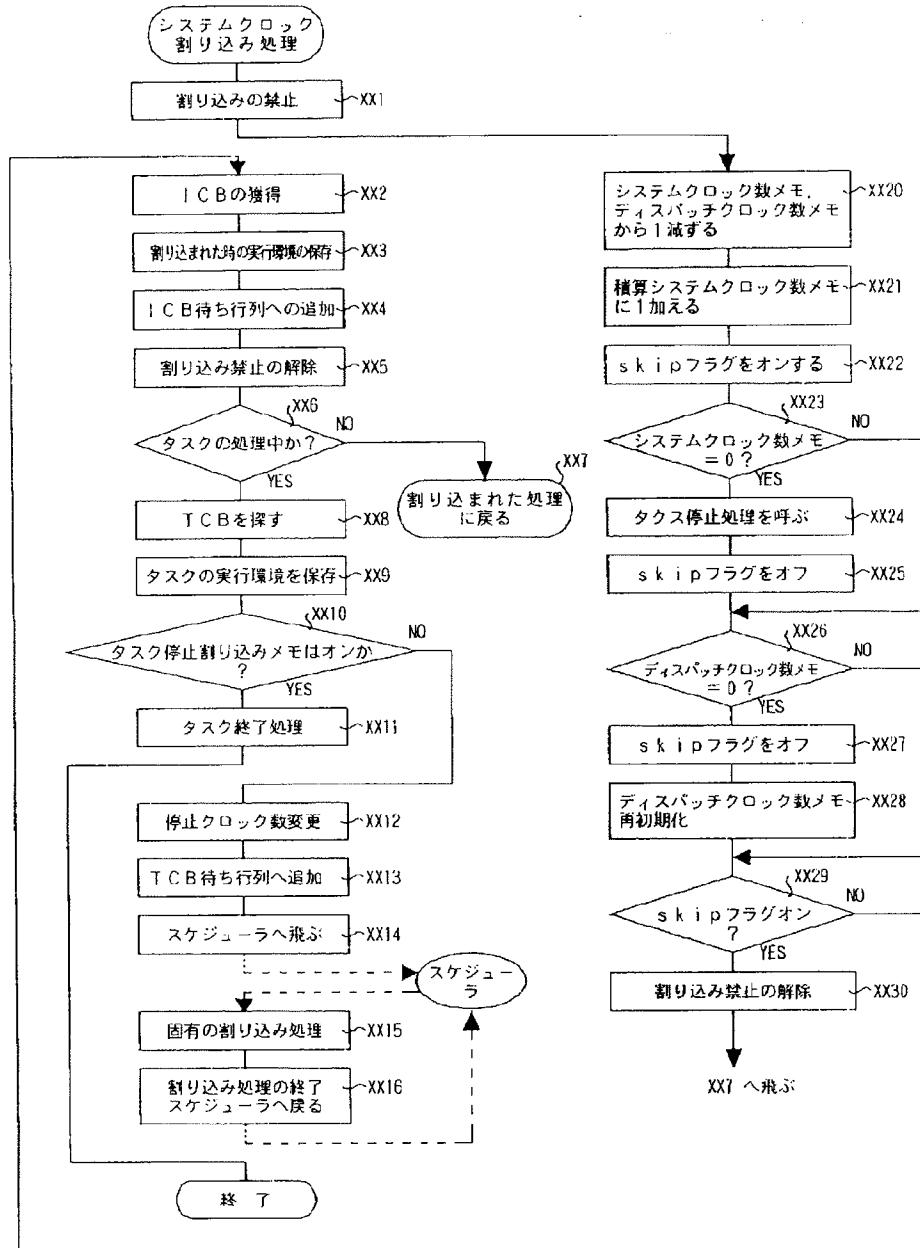
【補正内容】

【図71】

551  
走行時間判定許容度

552 走行割合判定許容度

上限	下限



特開平8-235004

【公報種別】特許法第17条の2の規定による補正の掲載

【部門区分】第6部門第3区分

【発行日】平成13年2月16日(2001. 2. 16)

【公開番号】特開平8-235004

【公開日】平成8年9月13日(1996. 9. 13)

【年通号数】公開特許公報8-2351

【出願番号】特願平7-205600

【国際特許分類第7版】

G06F 9/46 340

G05B 19/02

【F I】

G06F 9/46 340 E

G05B 19/02 T

【手続補正書】

【提出日】平成11年12月15日(1999. 12. 15)

【補正対象項目名】図63

【補正方法】変更

【補正内容】

【図63】

【手続補正1】

【補正対象書類名】図面

